Predicting Transaction Times

Eoin Carroll B.Eng. and Kieron Ellis B.Comm.

A Practicum submitted to University College Dublin in part fulfilment of the requirements of the degree of M.Sc. in Business Analytics

Michael Smurfit Graduate School of Business, University College Dublin

September, 2017

Supervisor: Dr. James McDermott

Head of School: Professor Ciarán Ó hÓgartaigh

Dedication

This work is dedicated to our friends and families.

Table of Contents

List of Figures

List of Tables

Acknowledgements

Thanks to David Talbot for his support and guidance.

Thanks to James McDermott for his mentorship.

The datasets used in The Practicum remains restricted however this document and the software developed may be shared.

Executive Summary

The Business Process Outsourcing department of Microsoft's Operations division processes work items in the form of transaction cases. These cases take differing lengths of time to process because each case is complex and must be processed manually. This leads to uncertainty as to whether they will be completed before key dates such as the end of financial quarters. Currently, Microsoft's management use the average case processing time to assist in allocating staff resources but this is a poor method of prediction because processing times vary so much on a case-to-case basis. With the roll out of a new transaction logging system, Microsoft identified the opportunity for the creation of a prediction system.

The objective of The Practicum was to provide a recommendation to Microsoft on how to predict the time required to process cases. For Microsoft's management team to consider the implementation of any recommendation, it would need an accuracy of at least 90% correct predictions within a tolerance of 4 days.

A detailed analysis of both Microsoft's business problem and data was conducted. Following this analysis, numerous data preparation steps and predictive machine learning models were optimised, tested and compared. The resulting model uses specific preprocessing steps such as filtering, transformation and variable generation to prepare the data for modelling and a Random Forest Regression algorithm to predict the time to process each case. This recommended model provides a significant improvement over the method of prediction currently employed by Microsoft and exceeds the target accuracy of 90% correct predictions within 4 days.

Figure 1 provides a comparison of the recommended model and average case processing time predictive accuracies. The recommended model offers a big improvement over the average case processing time predictions. The recommended model reaches the target accuracy of 90% correct predictions within a tolerance of just 2 days and reaches over 96% in 4 days, compared to Average Processing Time which reaches just under 28% in 4 days.

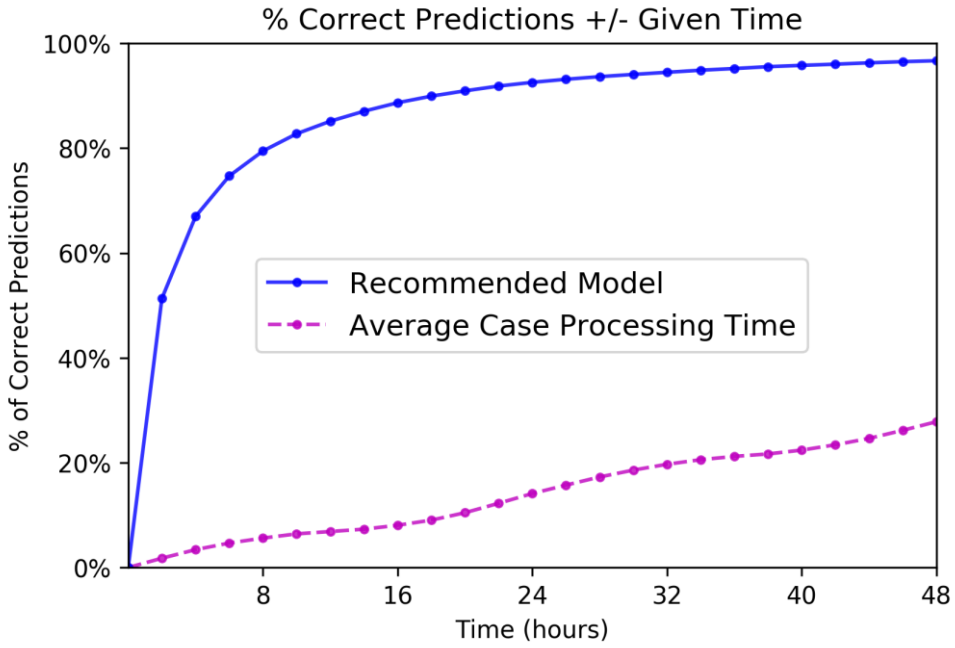

Figure 1: Comparison of recommended model and average processing time

The most predictive variable was the number of concurrently open cases at the time a new case was entered onto the system. The time remaining until upcoming key deadlines was also important. Surprisingly, variables directly recorded by Microsoft's case logging system were not very predictive of the time taken to process a case. This implies that the processing times for cases are a result of the time of the year and staff workload rather than individual case details such as their auditing requirements.

The recommended model uses only variables which are available from a case's inception which means that Microsoft can make predictions in real time for ongoing cases.

The results of several case study experiments revealed that the model will require a full year of data before it can be implemented. Further recommendations to Microsoft included following the CRISP-DM framework which is a project management technique for data mining.

In future iterations, the model should be updated to utilise two new features which were not available during testing. Firstly, seconds to the end of year should be included once a full year of data is available. Secondly, staff resource counts should be included to allow for more accurate predictions along with scenario analyses.

GitHub link: https://github.com/K-Ellis/Predicting-Transaction-Times

List of Important Abbreviations

MS: Microsoft

BPO: Business Process Outsourcing

SDM: Service Delivery Manager

COSMIC: Microsoft's CRM logging system

QTR: Quarter of a financial year

SEMMA: Sample, Explore, Modify, Model, and Assess (SAS project management)

CRISP-DM: Cross Industry Standard Process for Data Mining

B: Baseline

EN: Elastic Net

GBR: Gradient Boosting Regression

LR: Linear Regression

RFR: Random Forest Regression

R^2: R Squared Error

RMSE: Root Mean Squared Error

MeanAE: Mean Absolute Error

MedianAE: Median Absolute Error

Chapter 1 - Introduction

The ability to predict time requirements for a project or task is a strategic advantage for any organisation. Resources can be deployed more efficiently and timeline commitments are more likely to be met which can increase both customer satisfaction and profitability. It is becoming easier to implement prediction systems in the digital age where vast amounts of data are available along with open source machine learning libraries and an abundance of computing power. The design and experimentation of a predictive model for work item processing times forms the basis of this project and capstone report document, henceforth referred to as The Practicum.

1.1 Business Background

The Business Process Outsourcing (BPO) department of Microsoft's Operations division processes work items in the form of transaction cases. These cases take differing lengths of time to process which leads to uncertainty as to whether they will be completed before key dates such as the end of financial quarters. With the roll out of a new transaction logging system, Microsoft has identified an opportunity to develop a prediction system to assist in allocating resources to more effectively process transactions.

1.2 Objective

The objective of The Practicum is to provide a recommendation to Microsoft on how they could predict the time required to process new work items. This recommendation will be based around a methodology of how to obtain insights from their data to implement a prediction system. In addition to this, recommendations on how to improve prediction accuracy through additional data recording or further research will be included.

1.3 Data mining

Data mining is the process of obtaining useful information from databases. It is a branch of computer science that combines machine learning and statistics. Trends and correlations can be extracted from datasets and used to estimate future data points. The Practicum is a data mining project with the objective of modelling historic closed transaction cases so that a time to completion for new cases can be estimated.

1.3.1 CRISP-DM

The Cross-Industry Standard Process for Data Mining (CRISP-DM) is a data mining methodology developed by industry experts in the late 1990s to make data mining more accessible while sharing best practices (Shearer, 2000). In contrast to other models such as SEMMA, CRISP-DM does not specify a software platform for data mining and so it has been widely adopted.

The CRISP-DM methodology is an iterative process that includes six distinct phases. During each iteration, and as more background and insights are gathered, the understanding becomes deeper and the model becomes more advanced. The phases are shown in Figure 2 and are briefly described below. The CRISP-DM framework was utilised to great effect in The Practicum.

1. Business understanding: What does the business hope to gain from the project?
2. Data understanding: In what form, how reliable and how detailed is the data?
3. Data preparation: Clean and transform the dataset as necessary.
4. Modelling: Select and build a model using the prepared data.
5. Evaluation: Evaluate the results of the model.
6. Deployment: Plan, report and implement findings.

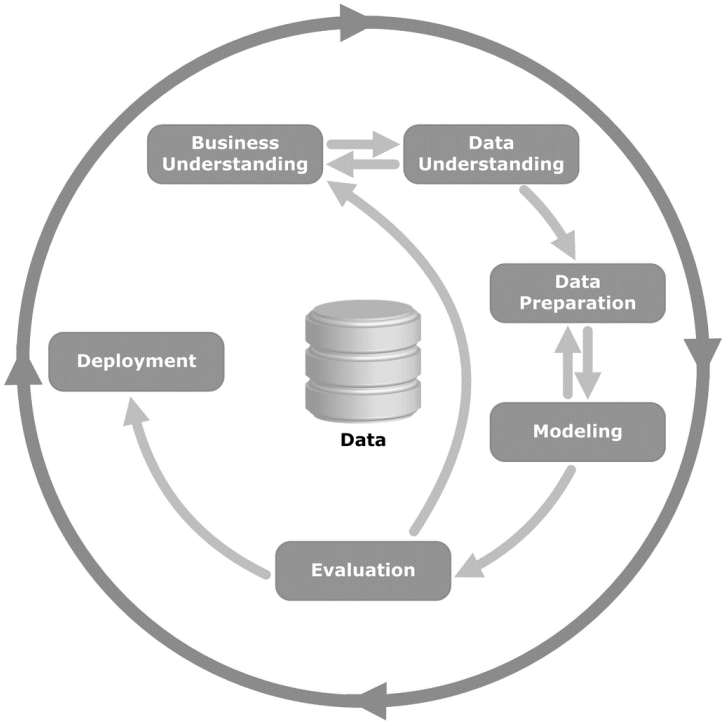

Figure 2: CRISP-DM process diagram (Wikipedia, 2017)

1.4 Process Mining

Process mining is the mining of data that is in the form of event logs where each log entry corresponds to a step that a case has taken within a process. While data mining can be used to examine the case as a whole, process mining can provide an end to end understanding of each step in the process. Both techniques aid process improvement projects such as business process management or value stream mapping (Van Der Aalst, 2012).

The Process Mining Manifesto describes process mining as midway between data mining and process modelling (IEEE Task Force, 2011). Software such as ProM is available for process mining, however the event log data provided for The Practicum is not robust enough. The Practicum sits between process mining and data mining. Therefore, a data mining framework with some process mining inspired aspects was adopted.

1.5 Practicum Layout

The remainder of The Practicum is set out as follows: Chapter 2: Business Understanding covers the business background, problem understanding and objectives. Chapter 3: Data Understanding describes the data that was available for The Practicum. Chapter 4: Related Work explores previous work that has been documented in this research area. Chapters 5, 6 and 7 describe the methodology used in preprocessing, modelling and evaluation. Chapter 8: Experimentation & Case Study and Chapter 9: Results & Discussion demonstrate the model's capabilities and captures important findings. The Practicum concludes with Chapter 10: Conclusion.

Chapter 2 - Business Understanding

Microsoft is a publicly traded company which means its earnings are made publicly available. This requires that Microsoft has its financial reporting in order by the end of each month, quarter and particularly fiscal year, to best showcase its worth.

Figure 3: Microsoft logo (Microsoft, 2017)

Intricate licensing agreement cases (agreements, credits, orders and queries) are sent to Microsoft by its customers. These tailor-made deals must be manually processed by Microsoft's Business Process Outsourcing (BPO) agents before the end of each month and quarter for them to be included in Microsoft's earnings reports and to meet customer deadline commitments. Microsoft's Operations Service Delivery Managers (SDMs) are responsible for providing the Microsoft leadership team with their predictions as to whether crucial BPO cases, which are currently queued, can be processed before the deadline.

With often $1 Billion in revenue coming through Microsoft Operations on the last day of a financial quarter, there is a huge amount of pressure on the SDMs to correctly predict how much resources are necessary for these cases to be processed in time. Currently, the average handling time is the only estimator being used to predict the workload.

Complicating matters, a single case can contain multiple processing transactions which must be processed before the case can be closed. As of now, Microsoft has no way of telling how many transactions will be contained in a case. Having considered this issue multiple times, Microsoft's vendor has concluded that there is no way to make predictions as to the number of transactions in a single case. However, a new customer relationship management (CRM) system, COSMIC, has been adopted by the BPO team and Microsoft's SDMs are hoping it contains enough granularity so that predictions into the handling time of orders can now be made.

Microsoft is a U.S. company and therefore many orders fall under the Sarbanes–Oxley Act (SOX) which requires additional compliance checks to be carried out. These orders typically

follow five stages: Ops in, Triage and Validation, Data Entry, Submission and Ops out. Data Entry takes place in Manilla and the four other stages are carried out in Dublin.

With a very large percentage of the total volume of orders coming in to Microsoft on the last days of the month and quarter, work rate increases and revenue based orders are given higher priority. Thus, seasonality effects have an impact on the handling times of cases.

The SDMs want to be able to make accurate predictions in real- time about whether they will be able to process all the queued and in-progress cases in time for the end of month and quarter deadlines.

To measure the handling time for orders, the total time between when a case was created and resolved is measured. The SDMs are interested in time taken from the customer's perspective, so internal holds are not excluded. However, customer and 3rd party holds are included in the total time taken.

When a case is first opened, it contains only a ticket number (i.e. a unique identifier) and a Created_On timestamp. This data is automatically stored on COSMIC (Microsoft's CRM system). As the case is processed, more and more data is filled in manually by Microsoft's' agents (with approximately a 15 minute's delay before this data is stored by COSMIC). Agents work alone on each case, manually filling in first several mandatory data fields and then moving on to the others. This means that initially when predicting the completion time for an ongoing case, no data, aside from the Created_On timestamp will be available from which to make predictions. Then, as the case is processed, more and more data will be available to make predictions from.

Microsoft did not have a specific prediction accuracy goal in mind when they originally put forward the project. When pressed for an indication of what would be useful to the BPO department, the response was that a solution which provides of greater than 90% correct predictions correct to within 4 days (+/- 48 hours) would be something that the department could implement with confidence. This represents the scenario of determining if cases in the system will be closed before the end of the month or quarter end - a review that is typically carried out 4 days prior to the deadline.

Chapter 3 - Data Understanding

Historical case data was extracted from Microsoft's newly implemented COSMIC records logging system and provided for The Practicum. Four datasets were shared over the course of The Practicum, each representing longer time periods as the more data was recorded and became available. The most recent dataset included all previous records and is used throughout The Practicum unless otherwise stated. This data is in the form of an excel workbook with 4 data sheets. Additional sheets which include the SQL queries used to obtain the data were also shared.

The 4 datasheets are:

1. vw_Incident – 50,193 rows and 69 columns
2. vw_HoldActivity – 36,409 rows and 16 columns
3. vw_AuditHistory – 94,928 rows and 23 columns
4. vw_PackageTriageEntry – 20,895 rows and 12 columns

The vw_Incident datasheet contains a list of incidents (an incident is another name for a case), with one case per row. The dependant variable for The Practicum is the total time taken to process a case, Time Taken. It is calculated as the time between when a case is created and resolved (these are denoted by the Created_On and ResolvedDate variables respectively). Each case has its own unique "TicketNumber" and may contain multiple transactions, however the transaction level details are stored in the three other datasheets.

Table 1 shows the most notable variables in the vw_Incident worksheet along with their purpose. A full list of variables, including the preprocessing used for each variable is included in Appendix 3.

Table 1: Notable variables in vw_Incident worksheet

Variable Name	**Variable Use**
TicketNumber	Unique Identification number for a case
Created_On	Created in COSMIC by BPO team
ResolvedDate	Date the case was resolved
Queue	The location of case at time of pulling data. Used to assign to speciality teams to be worked on

StatusReason	Users change the status. Outcome of case at time of pulling data. Online includes Cases which have been resolved at the time of pulling the data.
Priority	Case priority
ValidCase	Binary - if a case was valid
Program	Program type
SubReason	Lower level detail of Reason
CountrySource	Where was the case submitted from
CountryProcessed	Where is case being processed
SalesLocation	Where is sale being made to
ROCName	Regional operating centre name
IsGovernment	Binary
LanguageName	Language
sourcesystem	Where the case came from - free text field for user. Which Microsoft IT system was used
Source	Where the case came from - drop down list.
OLSRevenue	Indicates if Online Services Revenue associated with case
Revenutype	Type of revenue that MS will receive. E.g. immediate or future value
StageName	Stages SOX cases pass through. If not SOX, this stays as "OpsIn"
AmountinUSD	Amount in USD
IsMagnumCase	MS Sox tool to be used or not
IsSignature	If a customer signature is required
Complexity	Case complexity
Numberofreactivations	Number of reactivations
IsSOXCase	Binary: A legal document requirement

Unlike the vw_Incident datasheet, where one row represents a complete process instance (i.e. one case per row), the other datasheets store their data in a similar form to an event log. In an event log, one row represents an event (transaction) and multiple rows together make up a case. An event log contains multiple timestamped events linked together by common case ticket numbers so that each case is a sequence of events which can be ordered by their timestamps. The event log data was deemed not sufficient for predictions.

vw_HoldActivity contains the hold data for cases stored in vw_Incident. There can be multiple holds for one case, meaning the same TicketNumber can appear multiple times in the vw_HoldActivity datasheet. A hold is defined as a recorded time period where a case is not being worked on for various reasons such as awaiting customer documentation.

Vw_AuditHistory also uses a common set of cases. It tracks the movement of cases through different stages when the case is a SOX case. Approximately 47% of cases are SOX cases (not including cases for which there are Null entries for this variable). The time spent in each stage, as well as the old and new stage names, are recorded across multiple rows.

The vw_PackageTriageEntry datasheet has been disabled since the data was first provided for The Practicum and it will not be available to Microsoft in the future. As such, it was made clear by Microsoft to disregard it.

A full list of the variables in vw_Incident, vw_HoldActivity and vw_AuditHistory including the preprocessing step and reasoning for decisions made is included in Appendix 3: Methodology while the methodology section describes the preprocessing steps in detail.

3.1 Time Taken

Time Taken is defined as the time difference between when a case was created and resolved.

Figure 4 shows a histogram of the entire Time Taken distribution, with each of the histogram's bins corresponding to an 8-hour period. Cases resolved within the first 8 hours is by far the most common occurrence. Seen in both the Figure 4 and Figure 5 histograms are interesting periodic effects with large peaks separated by valleys.

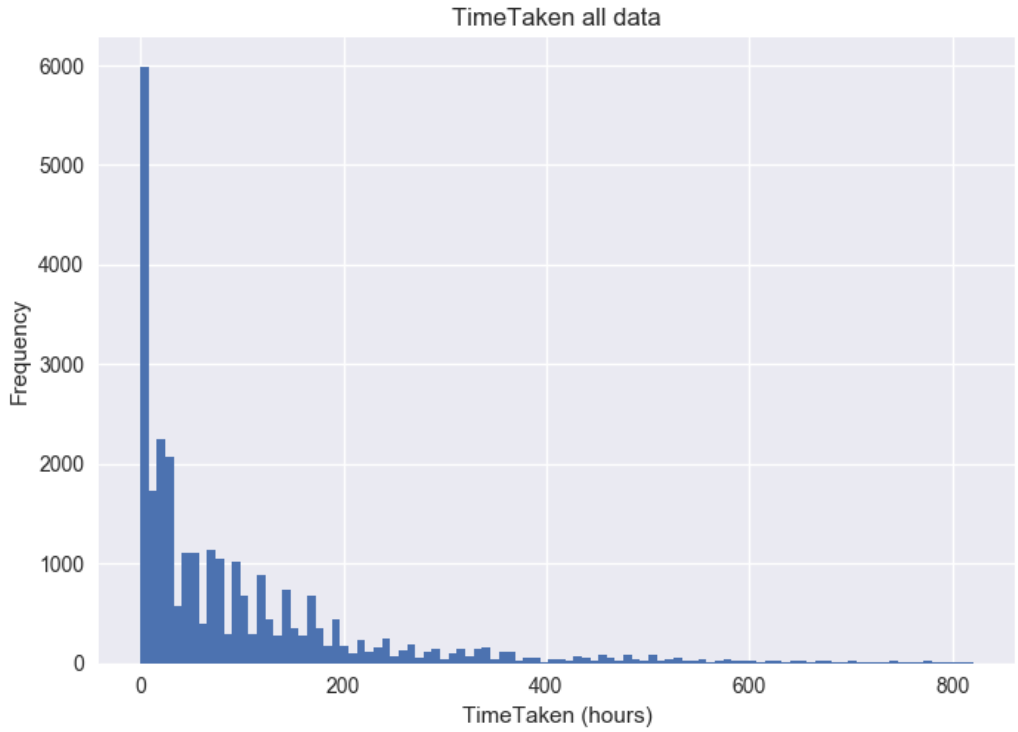

Figure 4: Time Taken histogram

Figure 5 shows a close-up of the Time Taken histogram for cases processed within 120 hours (5 days) with bins every 2 hours. Each of these peaks occurs for approximately 8 hours and each of these is separated by 24 hours. These 8-hour peaks have been coloured red. Either side of each peak, the Time Taken for cases drops off in volume. This is consistent with the 8-hour shift work which takes place at Microsoft.

Microsoft's agents in Ireland work on their cases only during their 8-hour shifts which start at 9am and finish at 5pm. Thus, the Time Taken to complete a case is unlikely to be outside of these times. It is unlikely that a case will be completed in 9 to 16 hours because if an agent starts work at 9am and immediately starts working on a case, after 8 hours the agent would be finished work for the day and no further work would take place outside the workplace. If a case is created at 9am, it is likely completion times are between 9am and 5pm (0-8 hours) or else the next day between 9am and 5pm (24-32 hours), et cetera. If a case is created at 5pm, then the likely completion times are between 9am and 5pm the next day (16-24 hours) or between 9am and 5pm the following day (40-48 hours), et cetera. Microsoft does have additional operations data entry staff in Manila but they are available for work 24 hours a day and therefore this has no impact on the shift patterns seen in these histograms.

Figure 5: Time Taken histogram <120 hours

Figure 6 shows boxplots for the time of the day cases were created. In blue is the mean value and in red is the median. As can be seen, the processing times for cases created between 1am and noon is longer than for cases created after 1pm.

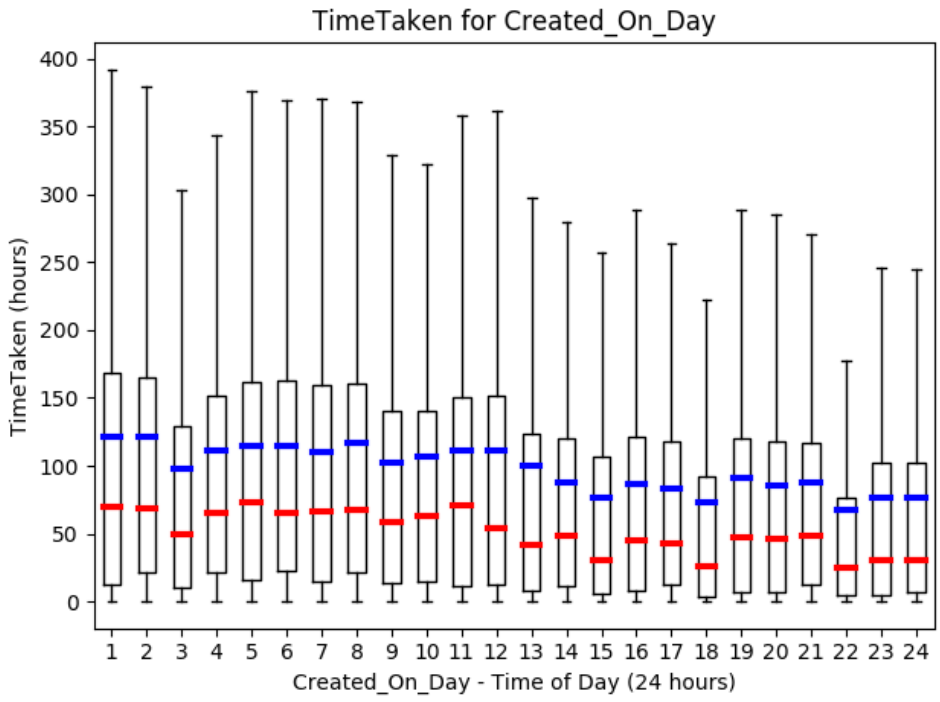

Figure 6: Created_On - time of the day boxplots

As seen in Figure 7, the time taken to complete a case seems to be roughly the same regardless of when a case was resolved, with the early hours of the morning being an exception to this rule. This could possibly indicate that the most complex cases are resolved throughout the day but not overnight. The Time Taken for overnight cases seems to be lower on average, i.e. they were resolved in a shorter amount of time than cases resolved during the day. This suggests that the more urgent or less complex cases are resolved overnight but the less urgent or more complex cases are resolved during the day.

Cases resolved at 8am and 9am have a longer average and median processing time. Cases resolved at this time were likely created on the previous working day which therefore would result in longer processing times.

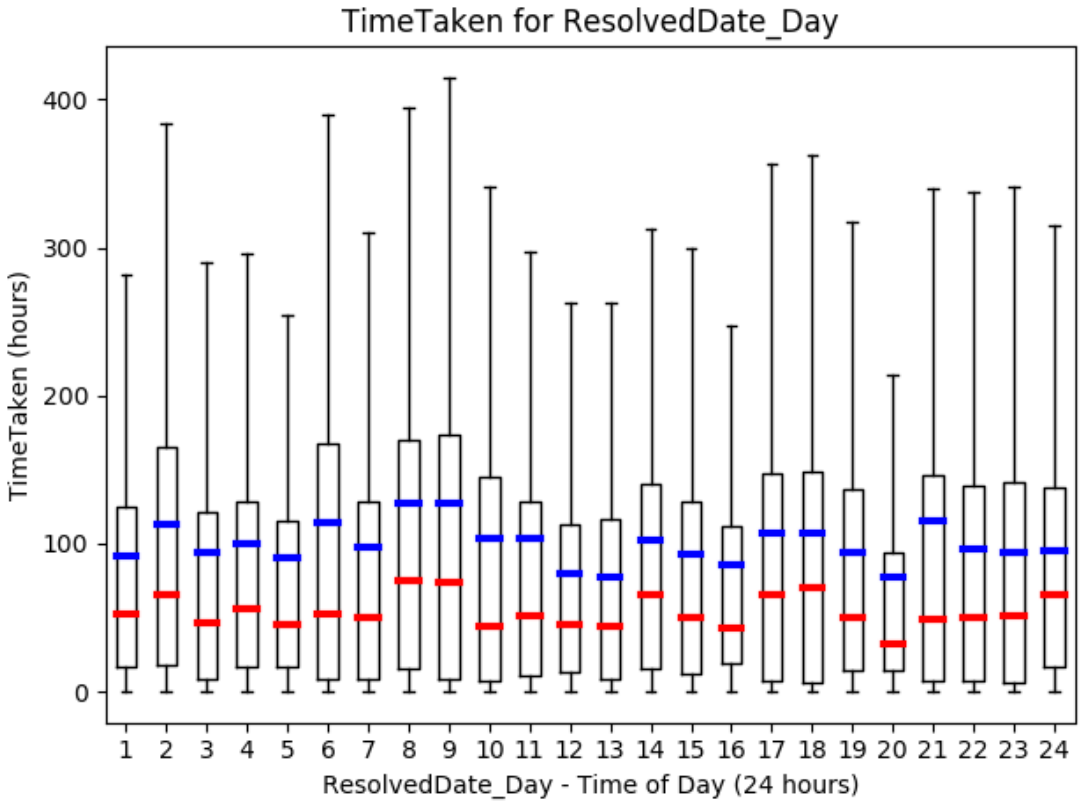

Figure 7: ResolvedDate - time of the day boxplots

3.2 Seasonality

Microsoft indicated that there is a seasonality aspect to the data because more urgency is placed on completing cases at the end of each month and quarter. A seasonality study was conducted to examine temporal patterns in the data. The Created_On and ResolvedDate variables were

converted to days of the week, month, quarter and year and then plotted against the time taken to complete each case.

As seen in Figure 8, cases created early in the week are resolved faster than later in the week. Wednesday, Thursday and Friday in this plot stand out because of their high large variability. On these days, cases are either resolved quickly, like early in the week, or else don't get resolved until after the weekend. This can lead to a lot of variability in the time taken for cases created on these days.

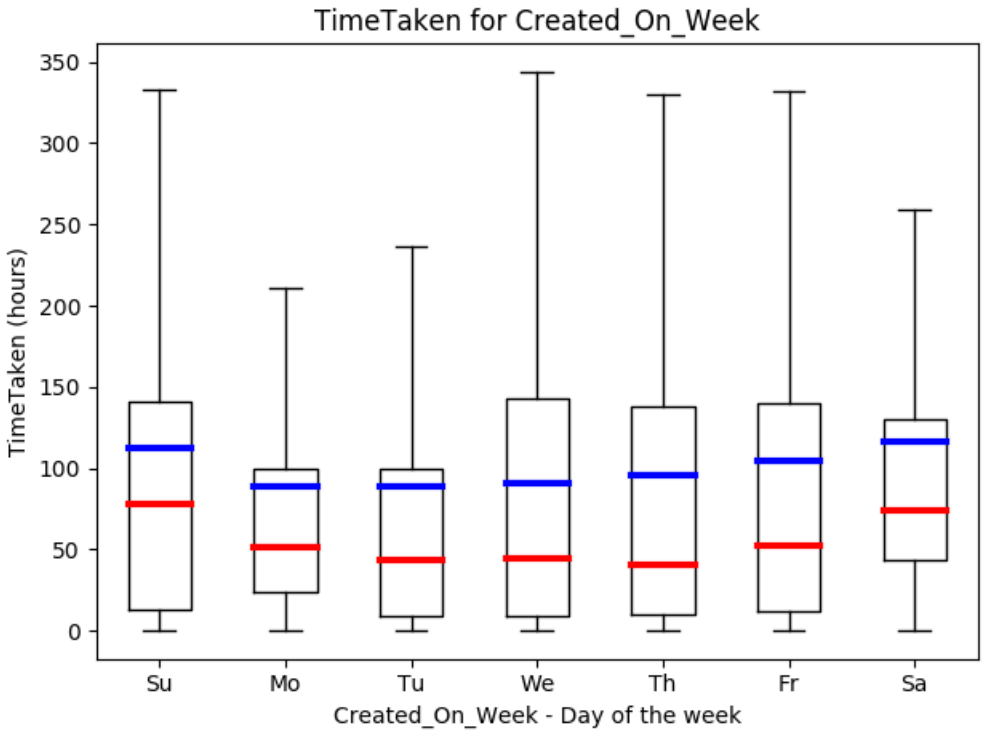

Figure 8: Created_On - day of the week boxplots

These findings raise an interesting decision point, i.e. Time Taken could be reformulated to include business hours only. However, for the purposes of The Practicum, Microsoft is only interested in the total time observed from their customer's point of view. In addition to this, a non-negligible amount of activity does take place outside of business hours which can be seen in later plots.

As can be seen in Figure 9, cases do get created on Sunday, which is always a non-business day for Microsoft. However, most cases are created during the week. As can be seen, fewer cases are created on Mondays and Tuesdays than on Wednesdays, Thursdays and Fridays.

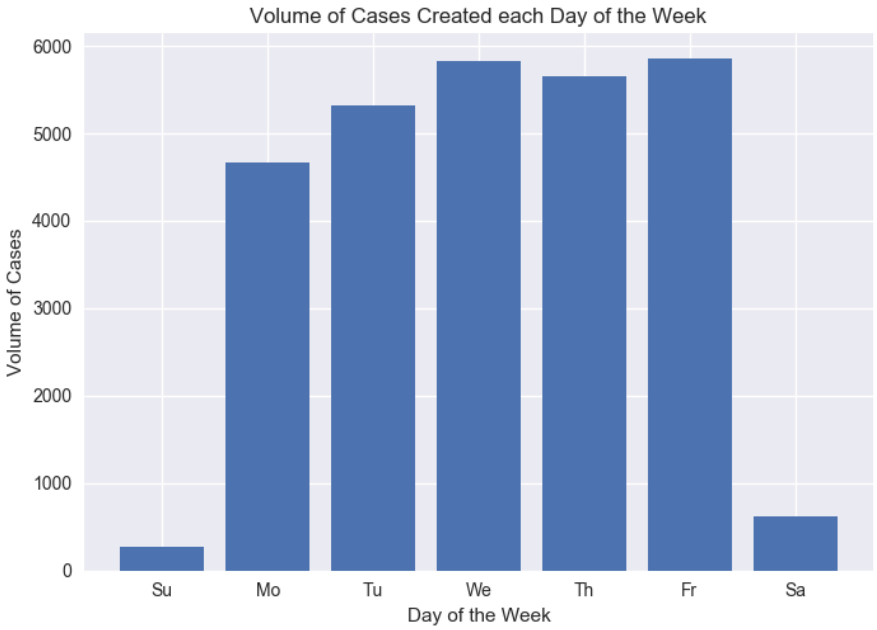

Figure 9: Volume of cases created each day of the week

As seen in Figure 10, cases resolved on Monday, Thursday and Friday had the lowest average and median processing times. There was also less variability in the Time Taken for cases resolved on Thursday and Friday because these cases were likely created on Monday, Tuesday or Wednesday.

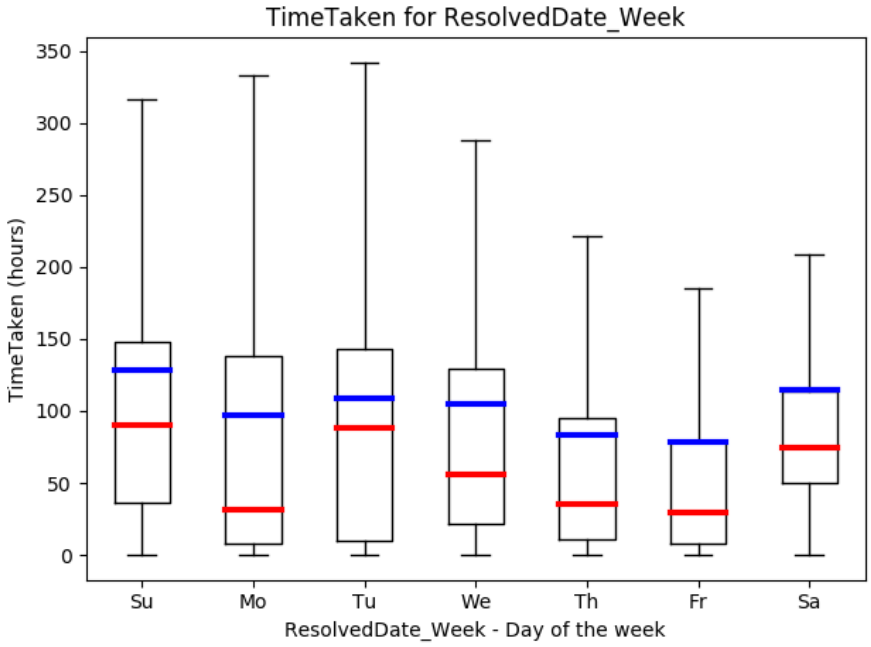

Figure 10: ResolvedDate - day of the week boxplots

As seen in Figure 11, most cases were resolved on weekdays, with fewer cases resolved on a Monday and a non-negligible number of cases get resolved over the weekend.

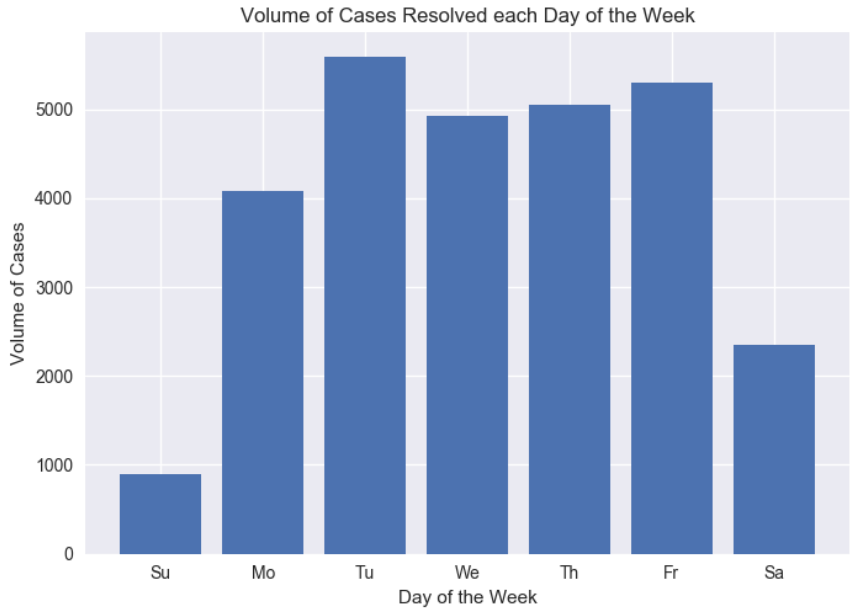

Figure 11: Volume of cases resolved each day of the week

As seen in Figure 12, by far the largest volume of cases were created in June, with May not far behind. It is clear from the data that the end of the fiscal year for Microsoft (June) is by far their busiest period which matches up with the Business Understanding. Very few cases were created in February because incomplete records were supplied for this month and few cases were created in August because this dataset was pulled midway through that month.

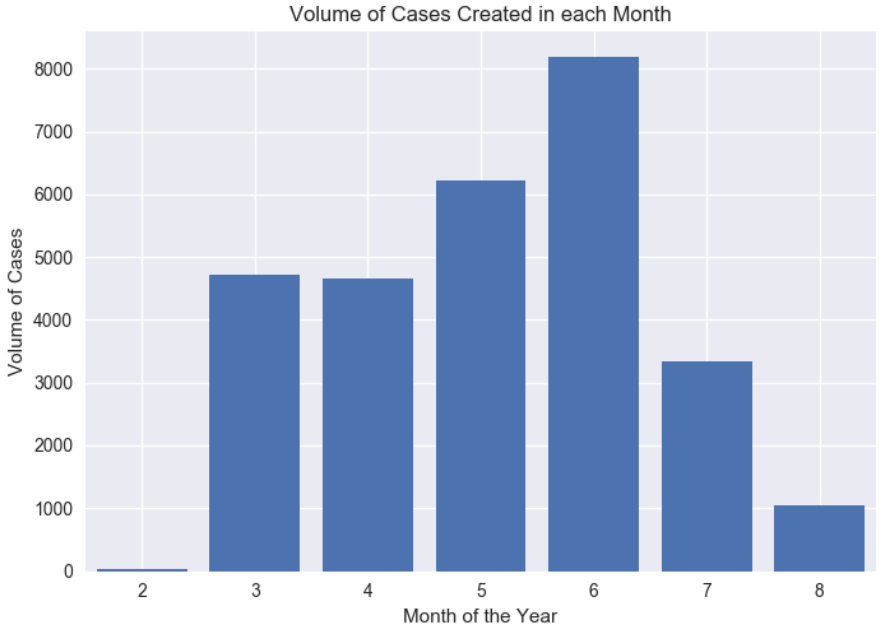

Figure 12: Volume of cases created in each month

Further Data Understanding, including analysis of the Time Taken for cases created and resolved on each day of the month, quarter and year is captured in Appendix 1.

Chapter 4 - Related Work

A review was conducted to search for literature related to the problem faced by Microsoft. Most related work was found to focus on event logs which are not directly transferable because of the nature of the available data, however, many of the ideas can be integrated in some way. The following section summaries other researcher's work which is related to The Practicum.

Reijers (2007) does not propose a tangible prediction model or technique, but instead provides an overview of the challenges faced by predicting the completion times for cases and makes recommendations based on these. Firstly, the author argues that any prediction must be highly accurate. This is because accuracy is the principal standard by which any technique will be judged by decision makers, practitioners and researchers alike. Secondly, a prediction must take place instantaneously because the longer the prediction takes, the less useful it is. Thirdly, any model must be easy to use for business professionals and managers so it can be incorporated into their work. This means that any manual input into the model must be kept to a minimum. And lastly, the requests made by the prediction model must not interfere with the process management system's performance or efficiency.

Van der Aalst et al. (2011) use process mining to predict the completion time of ongoing process instances (cases). Their approach, which was implemented in ProM (an open source tool for process mining), uses information learned from historical event log data to predict completion times for partial (ongoing) cases using an annotated transition system - a directed graph which is annotated with time information (Van der Aalst et al., 2010).

As seen in Figure 13, operational business processes are stored in an information system. Event logs are created from this system and following a process discovery phase, an annotated transition system is created which is used as a prediction engine to predict the completion times of running cases.

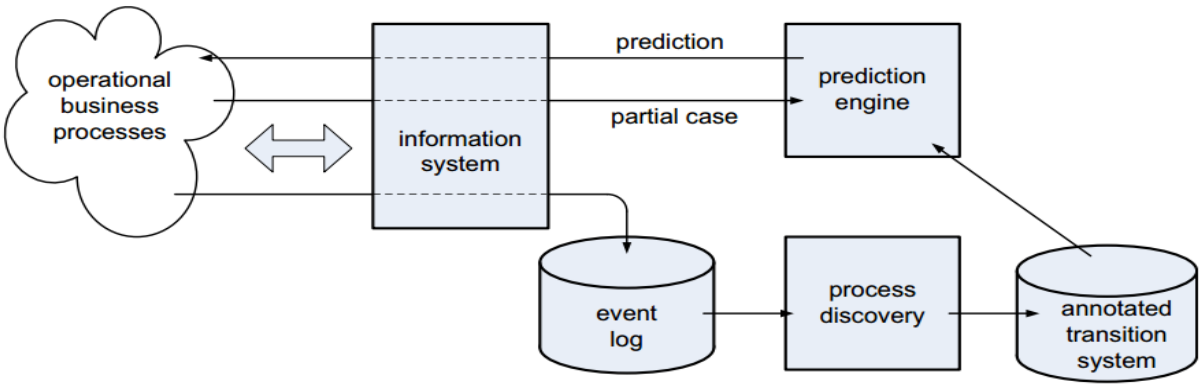

Figure 13: Overview of approach used by van der Aalst et al. (2001)

An event refers to a well-defined step in the process and each event relates to one case. It has a unique identifier and can contain other properties such as a timestamp, associated costs, the name of the corresponding activity, the name of the person who carried out the activity, etc. A trace is a sequence of events which correspond to an event and an event log is a set of traces.

Process discovery techniques are used to extract models from event logs.

A transition system describes all the possible behaviours that can cause a process to move from one state to another. It is made up of three components: the state space (all the possible states that a process can be in), the event labels and the transition relation (which describes how a case can move from one state to another). The transition relation is made up of three components: an initial state, an event label and a final state. All the possible outcomes for a case are given by all the viable walks in the transition system.

The goal of creating the transition system is to link each state with the thing we want to predict, i.e. the response variable. Measurements about the elapsed times, sojourn times and remaining times can be annotated to each state in the transition system, for example: the time to reach state A is 20 minutes, the time to complete from state B is 5 hours and the time spent in state C is 1 hour.

To use the annotated transition system for predictions, the set of measurements stored for each state needs to be converted to a single number i.e. if there are multiple time to complete measurements stored for state A, the average time to complete can be used. Other prediction

functions can also be used, for example taking the minimum or median time to complete for a state.

The authors of this paper have emerged as leaders in process mining and this publication has been identified as a definitive paper on the subject, amalgamating many research ideas prior to its publication. The ideas presented within have also formed the basis for further study, which is summarised shortly. The rest of this chapter is subdivided into "Backward" (from van der Aalst et al. (2011)) and "Forward" (from van der Aalst et al. (2011)).

4.1 Backward

Dongen (2008) set out to predict the completion times for administrative type cases based on past event logs. The motivation for this paper was to use non-parametric regression to estimate the remaining case time more accurately than the trivial solution of using average case times.

Non-parametric regression can adapt to different input parameter forms rather than fitting a strict model inclusive of all predictors. The data Dongen (2008) worked with was like the data used in The Practicum in that cases are entered into a system and more information is added over time as the case progresses. The result of this is that very little information is available initially.

Dongen (2008) uses an activity predictor, an activity duration predictor and a case attribute predictor to estimate the future shape of a case based on the information in the database at that point in time. Following these preliminary predictions, regression is used to estimate case completion times. It is possible that Dongen's (2008) methodology could have been applied to The Practicum, however, a simpler approach was taken instead.

Van der Aalst et al. (2010) present an online process mining framework for operational decision support implemented in ProM. Before their framework was presented, process mining had only been in used an offline setting for descriptive type analytics on historic event log data. Their framework can be used to predict remaining processing times and recommend activities which will minimize the expected processing times and costs for ongoing process instances. It can also be used to check for deadline violations because by feeding a running case into the process model, an alert can be made if its observed behaviour deviates from the behaviour used to fit the model. To predict the completion times of running cases, the partial trace (incomplete case)

in question is fed into the transition system (which is annotated with time information) to identify its current state and then the average or the median of the time to complete annotations for that state is returned as the predicted remaining time to complete for that case.

Van der Aalst et al. (2007) successfully use business process mining, implemented in ProM, to analyse real-life invoices sent from various subcontractors and suppliers to the provincial offices of the Dutch National Public Works Department. The authors took these invoices, created an event log and automatically constructed a process model which described the behaviour seen in the log. Their efforts were purely descriptive and prescriptive but not predictive. For example, they discovered irregularities in the event log which slowed down certain case's completion times and used informed management about these insights so they could be investigated further.

Günther and van der Aalst (2007) propose a process mining approach, which is implemented in ProM, to extract process models from unstructured event logs. Unlike traditional methods, which don't make a distinction between important and unimportant data in the event log, their approach uses Fuzzy Mining to retain the highly significant behaviour, combine the less significant but highly correlated behaviour and remove the less significant and uncorrelated behaviour found in the event logs.

Günther and van der Aalst (2007) argue that traditionally process mining has made assumptions about the event log's accuracy and credibility that in real-life is often not the case. Historically, further assumptions have been made about the existence of a flawless solution which can be found. However, in real life, work flow management system processes are not carried out flawlessly and so less structured processes are likely to be recorded in the event logs. Process models which assume perfect structure exists in these unstructured logs result in "spaghetti" process models being made.

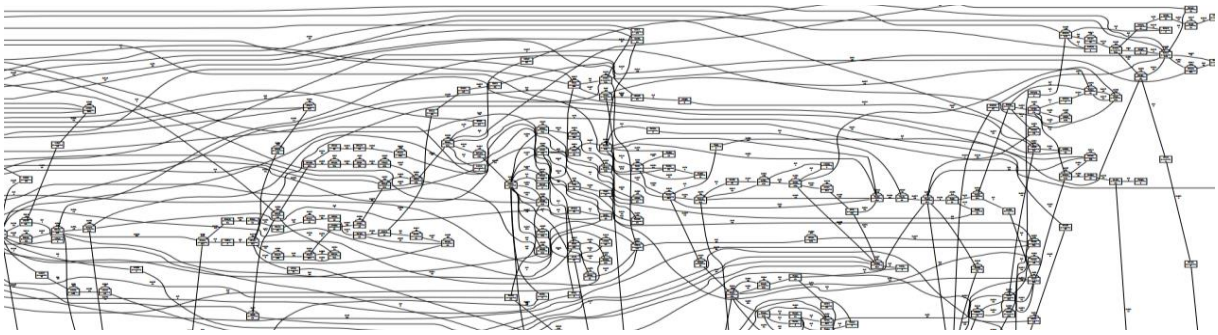

Figure 14: Günther and Van der Aalst's (2007) example of "Spaghetti" process model

Shown in Figure 14 is a typical "spaghetti" process model where the edges on the graph between the activity nodes are the "spaghetti". Günther and van der Aalst (2007) maintain that these "spaghetti" models are not incorrect, but rather it is extremely difficult to derive any meaningful understanding from them. Thus, the authors argue that a more simplified approach (as shown in Figure 15) to modelling real life unstructured event logs that provides a higher-level view of the process, and removes, combines and emphasises behaviours, can result in more meaningful insights and analyses being made.

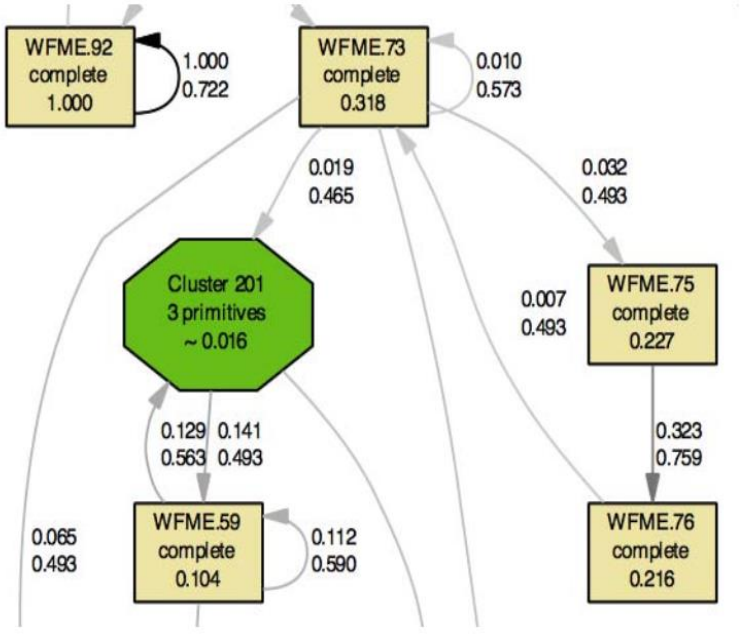

Figure 15: An excerpt from Günther and van der Aalst's (2007) simplified process model

Nakatumba and van der Aalst (2009), using ProM and Linear Regression analysis, explore the relationship between workload and service times. They argue that there exists a great discrepancy between "traditional" models and reality caused by incorrect assumptions being made about the real world. Their paper investigates one such assumption, i.e. the assumption that work rate is a constant speed. The authors contend that the "Yerkes-Dodson Law of

Arousal", originally developed by Yerkes and Dodson (1908) and visualised Figure 16, holds true. This "Law", from an operations management perspective, suggests that as a deadline approaches workers will increase their work rate and conversely if the time pressure is too great, their performance will decrease.

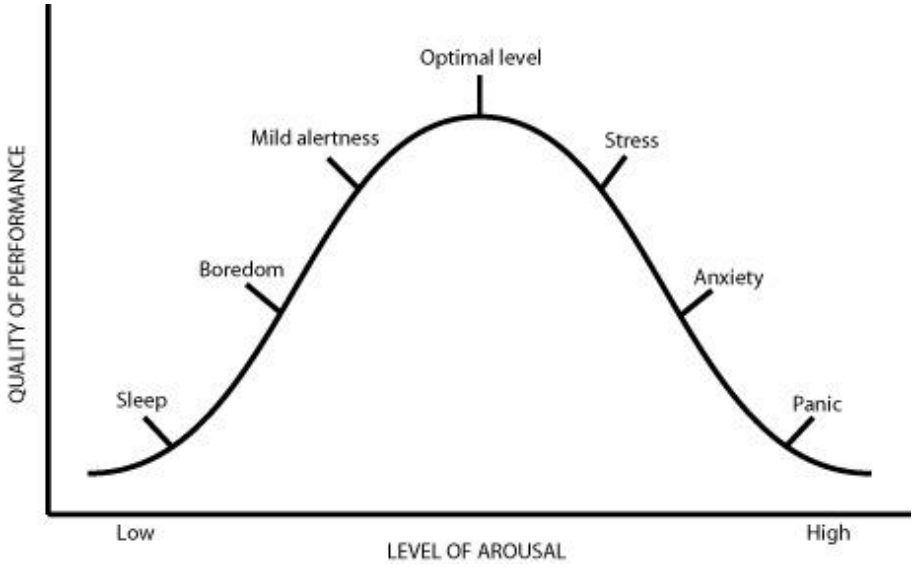

Figure 16: The "Yerkes-Dodson Law of Arousal" (Yerkes and Dodson, 1908)

Nakatumba and van der Aalst (2009) show empirically, using Linear Regression applied to a real-life event logs supplied by a Dutch municipality, that this relationship between workload and work rate exists. However, they contend that to truly capture the inverse U-shaped curve seen in Figure 16, a more sophisticated regression technique would be needed.

Schonenberg et al. (2009) present a framework for modelling real-life processes whereby the probability of a state in a transition system moving to another state depends on the history of those transitions the event log, i.e. their stochastic process model is annotated with the probability distributions of transitions based on their historic movements.

Van der Werf et al. (2008) present an integer linear programming (ILP) algorithm implemented in ProM to answer the problem of process discovery. The objective function they sought to minimize is defined such that states with fewer inputs and more outputs are favoured. They describe their results as promising, with their processing time scaling sub linearly with the size of the event log used. However, they contend that not all their constraints for structural properties lead to feasible solutions, and with a running time of 15 minutes (on a 2008 "standard" desktop PC) for an event log containing 1000 cases and 25 transitions, integer linear programming may not be the most efficient solution to the problem of process discovery.

4.2 Forward

The area of process mining has continued to grow since the publication of van der Aalst et al. (2011). Also in 2011, the Process Mining Manifesto was released by an IEEE Task Force on Process Mining. The task force was established in 2009 and consisted of industry experts including van der Aalst. The manifesto aimed to promote process mining as a tool by setting down guiding principles so that those interested in process mining could have a starting point for research and an easy to implement methodology.

Van der Aalst worked with Pika et al. (2012) to explicitly explore the risk associated with a process overrunning and missing deadlines. Their problem definition was similar to The Practicum because the expected closing time of a case is unknown. Pika et al.'s (2012) methodology was pattern identification using a process mining framework. It was implemented in ProM and tested on an event log case study.

Van der Aalst (2012) outline process mining opportunities for businesses and summarised the Process Mining Manifesto guiding principles. This paper is less technical than van der Aalst (2011) and so is more accessible for newcomers wishing to explore the process mining domain.

Lakshmanan and Khalaf (2013) provide a comparative analyses of different process mining algorithms and present a guide for which algorithm to select based on the user's available data. The authors categorise the different data types into unstructured data such as emails and phone calls, and semi structured data where case-oriented processes are captured in some way. They then define the steps that cases could take in a process: parallel tasks (completed simultaneously and in any order), loops (revert to earlier state), gateways (decision points where process flows fork), invisible/unrecorded and duplicate tasks (recorded in more than one location). Lastly, they describe a method for selecting which of five algorithms to use based on data type: alpha, fuzzy miner, flexible heuristics miner, genetics miner and the two-step approach. The data for The Practicum falls very much into the semi-structured category however the algorithms considered by Lakshmanan and Khalaf (2013) are not directly transferrable since The Practicum data is not suitable to be transformed into only an event log.

Rogge-Solti and Weske (2013) present a predictive approach for outstanding processing times based on anticipated next events using petri nets. Petri nets are directed graph models that represent discrete event transitions between statuses. The authors present an algorithm

implemented in ProM and applied it to a case study. The authors claim to have improved on previous approaches which predict based on time elapsed after the last event rather than expected events which have not yet occurred. The prediction of the future states of a case is important for The Practicum however an implementation using ProM is not applicable.

Business process management is an area of operations research study that aims to map and improve business processes. Van der Aalst (2013) survey the latest business process management thinking and provide many cases where process mining would be extremely beneficial in modelling business processes. The Practicum also aims to model a current state process so that any insights made can be used to manage the business's processes going forward.

Ceci et al. (2014) use sequential pattern mining to predict the completion times of processes. Their novel approach includes applying regression techniques to partial models. Polato (2014) also aimed to predict the time remaining in a process by using the available information to build a regression model. Both Ceci (2014) and Polato (2014) calculate the probability of next events using the available data for a case and then estimate the remaining time using a regression model. Polato (2016) continued this line of research and presented the business case for needing forecasts to manage service level agreements. Polato (2016) also included a method of prediction that works with constantly evolving processes - an important consideration for The Practicum.

One final publication worth noting was Riekert et al. (2017). The authors created a predictive model for the duration of surgeries in hospitals using regression. While the process steps were far more rigid than for The Practicum, the approach taken is somewhat similar. Riekert et al. (2017) used feature selection such as the time of day or day of week that a surgery was started on to create predictive models from patient data. The authors were able to improve on previous predictions for estimating time to completion.

As we have seen from related research and case studies, process mining is an important discipline area for The Practicum, however, a Data Mining regression model will be used due to the nature of available data. Ideas from the works presented above have been built on and implemented into The Practicum.

Chapter 5 - Preprocessing Techniques

This chapter describes the preprocessing techniques used on the dataset. The following sections are laid out in order of increasing complexity and reflect the order in which experimentations were carried out.

5.1 Data Filtering

Domain knowledge was used to remove entries which were outside the scope of The Practicum from the dataset. Microsoft specified that the data should be filtered to include a valid ResolvedDate, Program type - Enterprise, StatusReason - not Rejected and ValidCase - True. Cases were also filtered to only include English language cases.

A filter was placed on the IsSOXCase variable to remove entries with no information of SOX status. This decision was taken to reflect Microsoft's indication that SOX status was important and since the SOX status was only recorded from mid-March onwards, only entries prior to this date were excluded from modelling.

5.2 Time Taken

The dependant variable for The Practicum, Time Taken, was constructed by taking the time between the Created_On timestamp and the ResolvedDate timestamp. The result was saved in seconds, the smallest granularity available, to prevent unnecessary discretization error.

Once the Time Taken variable was included, the dataset was passed through another filter: entries where the Time Taken was greater than three standard deviations away from the mean were judged to be outliers and were removed.

The data is slightly right-censored because cases which were created towards the end of period when the data was collected had to be resolved quickly to be included in the dataset. This slight sampling bias means that cases which are resolved quickly are more likely to be represented in the dataset. However, after preprocessing, the final input dataset for the main experiments contains 28,204 entries gathered between February and August and the highest volume of cases coming into Microsoft was during June (the end of their fiscal year), so this small sampling bias should not have a large effect on any results obtained.

5.3 Data Transformation

5.3.1 Queue Variable

The Queue variable represents the location of a case at the time of pulling data and is used in assigning the cases to specific teams. The values of the Queue variable were combined based on the BPO team's Operating Centre name which reduced the number of Queue values from 82 to 8.

5.3.2 Country Variables

Three variables that used country names as values were transformed to use the continents each country is in. This was done to minimise the total number of values for each variable.

5.3.3 One-hot Encoding

Categorical variables including the Queue and Country variables were one-hot encoded. This technique creates a new binary column for each unique categorical value in a variable.

5.3.4 Ordinal Variables

The following categorical variables were judged to be ordinal variables: Priority, Complexity and StageName. These variables were transformed into numeric variables with sequential numeric values.

5.4 Imported Data

HoldDuration and AuditDuration were calculated by summing together the hold times from the vw_HoldActivity and vw_AuditDuration sheets for a TicketNumber and appending these times onto the vw_Incident sheet for that same TicketNumber.

Additionally, the AssignedToGroup and HoldTypeName variables from the vw_HoldActivity sheet were one hot encoded for each case, summarised and included for each TicketNumber also contained in the vw_Incident datasheet.

5.5 Generated Variables

Variable generation is the process of replacing or adding new variables inferred by existing variables.

5.5.1 Seconds left to key deadlines

The following variables were created by taking the time between the Created_On timestamp and the remaining time per day, month, quarter and year respectively.

1. Seconds_left_Day
2. Seconds_left_Month
3. Seconds_left_Qtr
4. Seconds_left_Year

Shown in Figures 17-20 are histograms for each of these variables.

The Seconds_left_Day variable is calculated as the number of seconds left from when a case is created until the end of the next business day. This is 5pm on weekdays, except for the last three working days of the month. On the third last and second last working days of each month the end of the business day is 11pm, and on the last working day of the month the end of the business day is 8am the following morning.

In Figure 17 it can be seen that the largest frequency of cases are created right before the end of the business day and a large amount of cases are created within 10 hours of the end of the business day. After 24 hours, very few cases are created as this period refers to weekends, during which agents generally do not work.

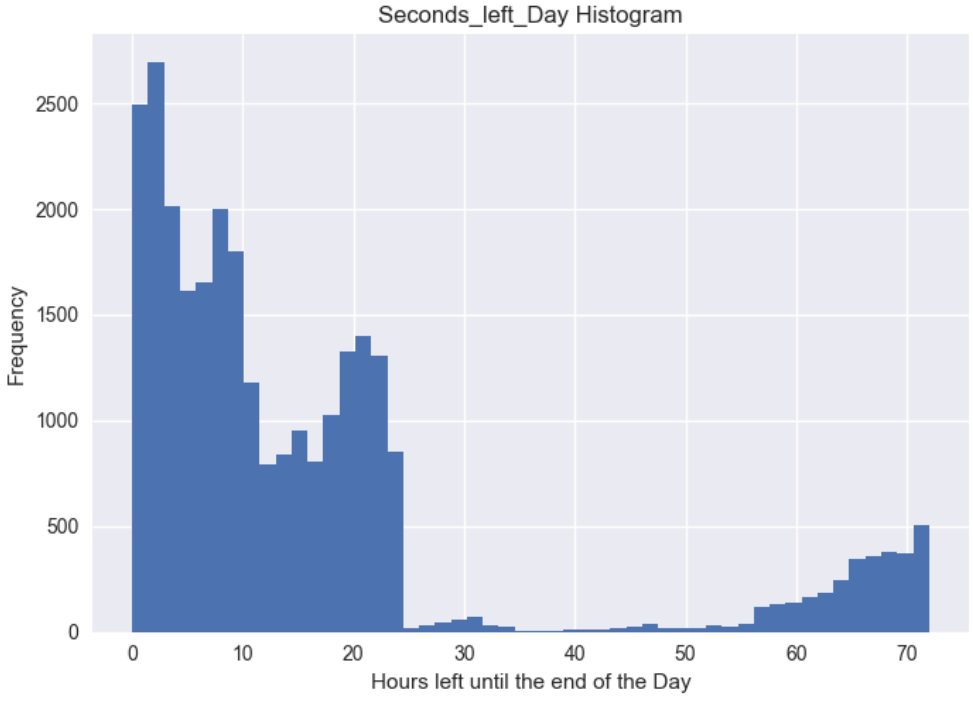

Figure 17: Seconds_left_Day histogram

Figure 18 shows the histogram for the Seconds_left_Month. This is defined as 8am on the morning following the last business day of each month. Each bin corresponds to 20 hours. It is clear that the largest frequency of cases are created during the last 100 hours of the month (~4 days) which almost corresponds to the last working week of each month (120 hours). A periodic effect can be observed where there is a noticeable peak lasting approximately 1 working week (120 hours) and each is separated by short valley which matches up with the number of hours in a weekend (48). Most cases are created during each working week, and more cases are created towards the end of the month.

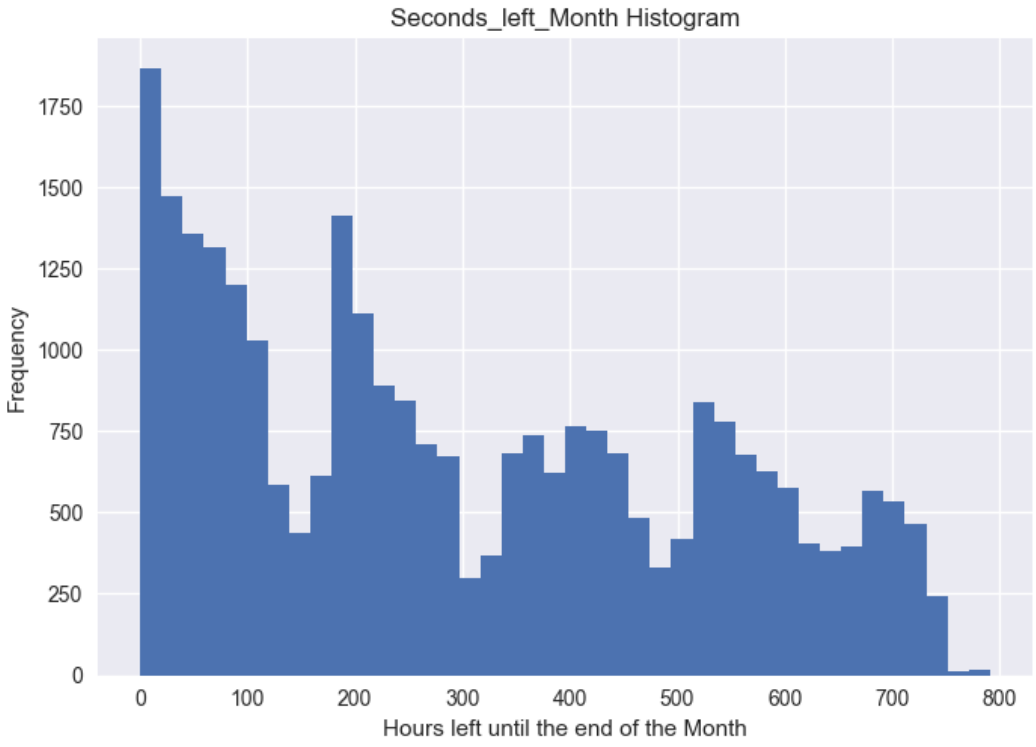

Figure 18: Seconds_left_Month histogram

Figure 19 shows the histogram for the Seconds_left_Quarter. This is defined as 8am on the morning following the last business day of each quarter. It can be seen that the largest frequency of cases are created at the end of each quarter. Two smaller peaks can be observed at around 700 hours (29 days) and 1600 hours (66 days). These correspond with the end of the first and second month in each quarter. It is very apparent that the end of the quarter sees much more cases created than any other time in the quarter.

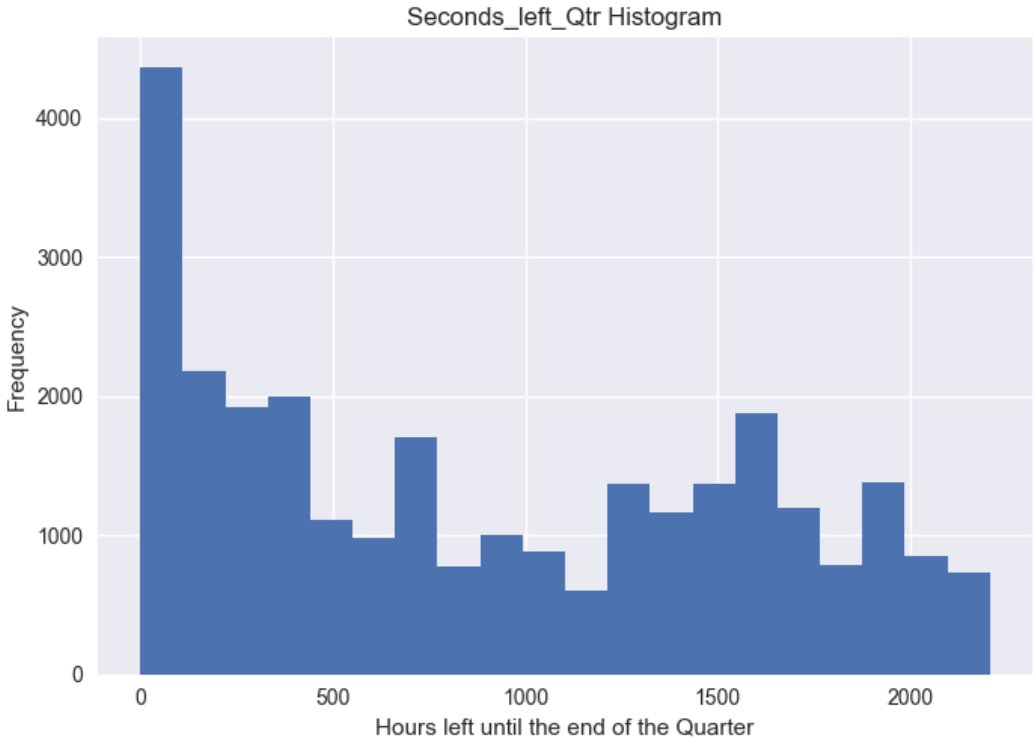

Figure 19: Seconds_left_Qtr histogram

Figure 20 shows the histogram for the Seconds_left_Year. This is defined as 8am on the morning following the last business day of the fiscal year. Again, the smallest values for this variable have the largest frequency. Four interesting peaks can be observed: starting from the left, the first peak corresponds to the end of the fiscal year (June), the next peak to the end of May, the third peak to April and the fourth peak to the end of the third quarter (March). The peak for March is larger than April's peak, which is unsurprising seeing as March is the end of a quarter. However, the peak for May is larger again. This implies that the last quarter of the year sees a larger volume of cases created than other quarters.

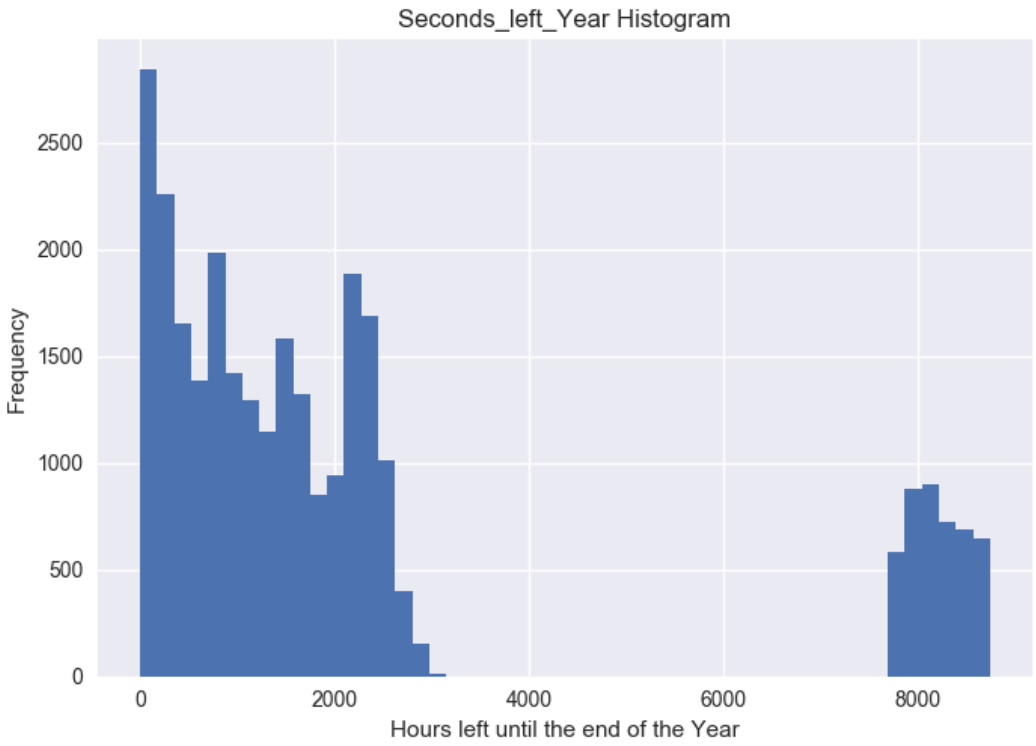

Figure 20: Seconds_left_Year histogram

The gap in the middle of Figure 20, between around 4,000 and 6,000 hours, corresponds to the ~6 months of data (September to February) which had not been collected by Microsoft's new CRM system. This is because this new system went into operation at the end of February and so a full year of data has not been collected. The data points at around 8,000 hours correspond to data collected after the end of the fiscal year (this is the most recent data collected in July and August). Here, at the beginning of the fiscal year, there are far fewer cases created than in the final months of the fiscal year.

5.5.2 Measures of Workload

The following variables were constructed by comparing the Created_On and ResolvedDate timestamp for each case with other entries in the database.

1. Concurrent_open_cases
2. Cases_created_within_past_8_hours
3. Cases_resolved_within_past_8_hours
4. Rolling_Mean
5. Rolling_Median
6. Rolling_Std

Concurrent_open_cases is defined as the total number of ongoing cases at the time each case was created. In Figure 21 it can be seen that there is not too much difference in the distribution of this variable between 0 and ~800 concurrently open cases, but greater than ~800 concurrently open cases is far less likely to occur.

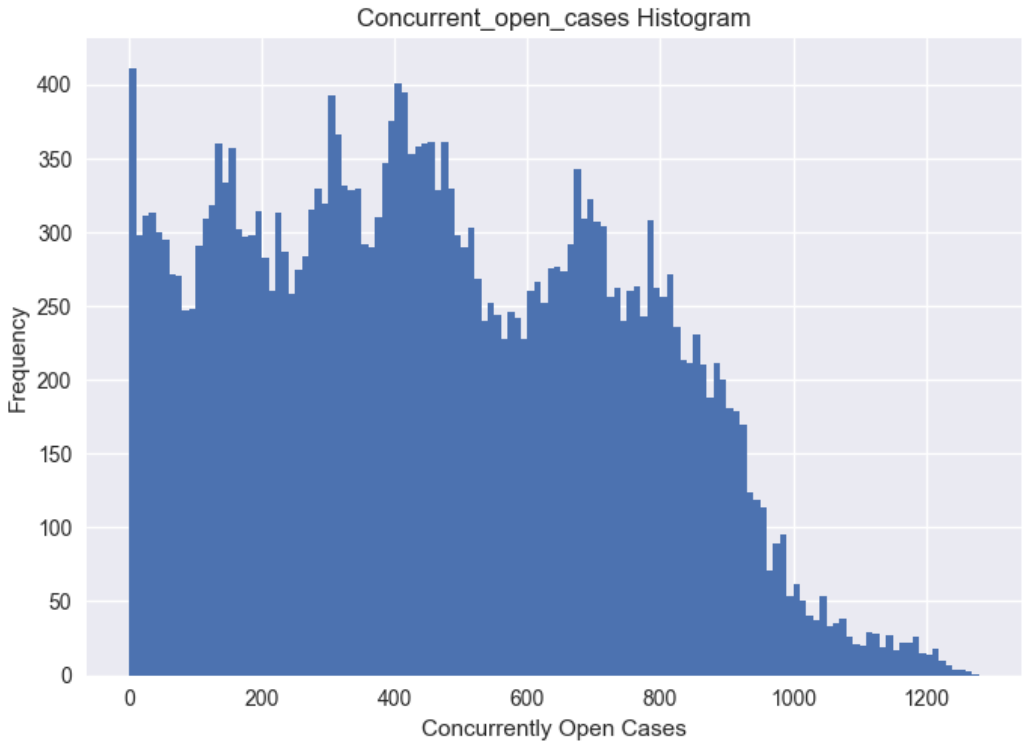

Figure 21: Concurrent_open_cases histogram

Figure 22 shows the average Time Taken for each case compared to how many concurrently open cases there are. A vertical standard deviation error bar is shown in blue every 10 data points. As will be shown later in The Practicum, this variable is by far the most predictive variable for Time Taken, so unsurprisingly this plot is very tidy and easy to interpret.

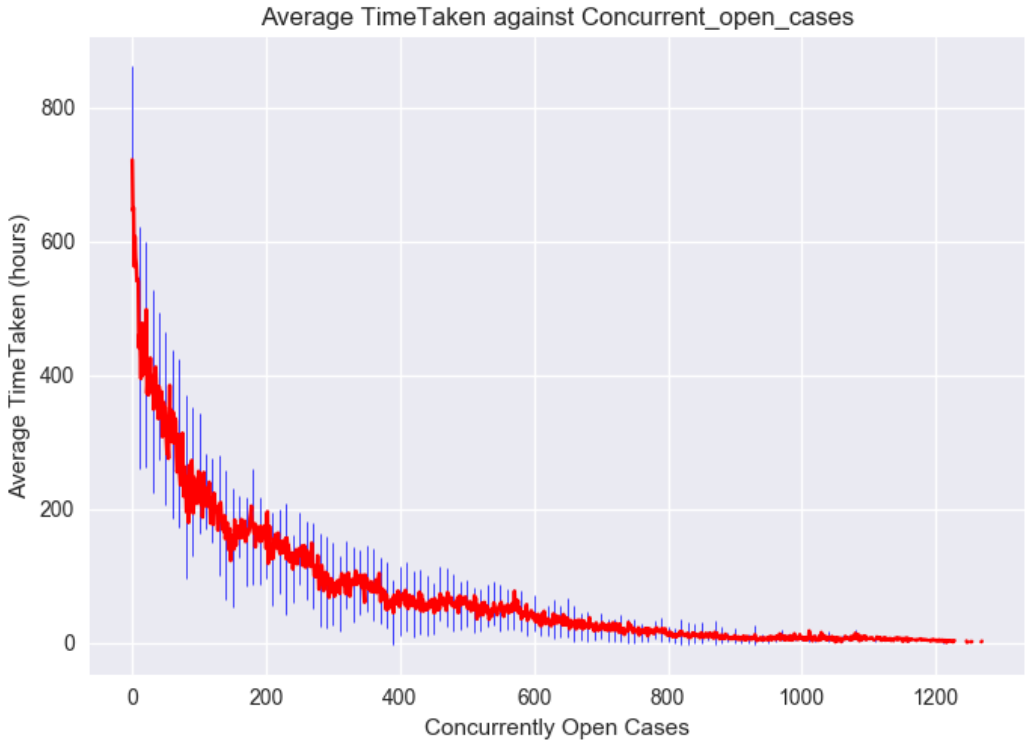

Figure 22: Time Taken against Concurrent_open_cases

As the number of concurrently open cases increases, both the average and standard deviation for Time Taken decrease. When there are fewer concurrently open cases, both the average and standard deviation for Time Taken are larger.

This is good news for Microsoft because the period they are most interested in predicting Time Taken values for (i.e. the last 4 working days of each month and quarter), has the largest frequency of cases created (as was shown in Chapter 3: Data Understanding). This means that these high-pressure periods are more likely to have more concurrently open cases and therefore are more likely to have a smaller average Time Taken. But also, the error for the average Time Taken for these periods is also smaller, meaning that there is likely to be less error in predicting Time Taken values during these times. As will be shown later in The Practicum, the start of each month has the largest predictive errors and the end of each month has the smallest, so this assumption holds.

Plots showing the average and standard deviations for Time Taken against other variables can be seen in Appendix 2. These tended to be chaotic and did not show clear correlations.

Figure 23 shows the histogram for the Cases_created_within_8_hours variable. There is likely to be between ~50 and ~100 cases created within 8 hours of each case. The Cases_resolved_within_8_hours histogram (not shown), was very similar to the Cases_created_within_8_hours histogram.

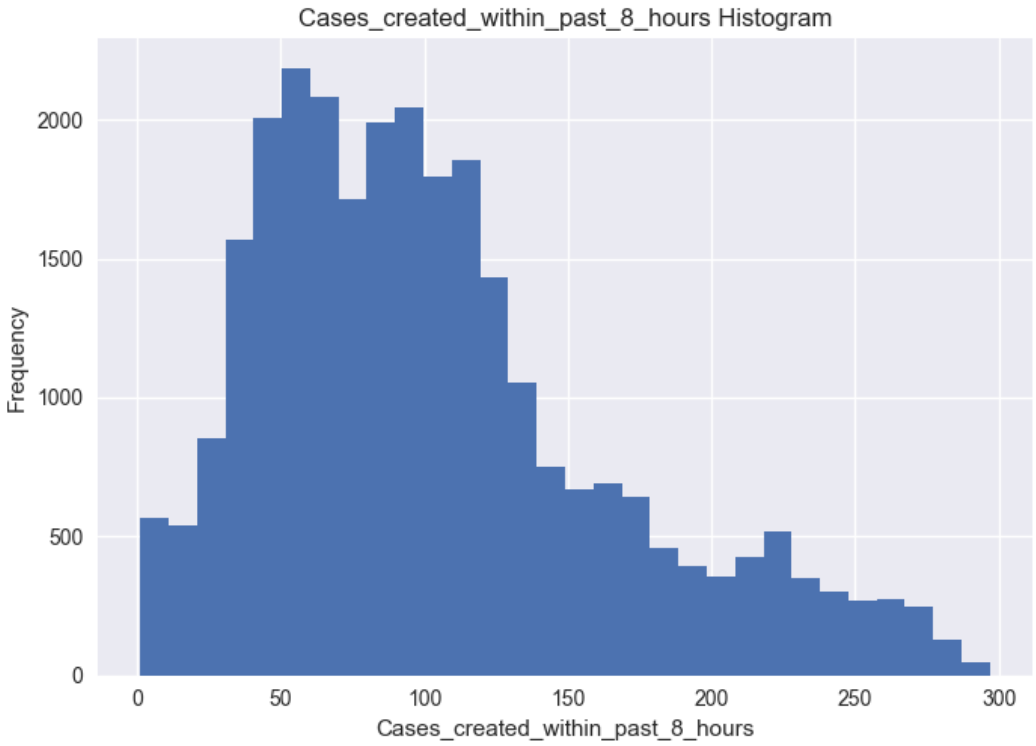

Figure 23: Cases_created_within_8_hours histogram

The Rolling Mean, Median and Standard Deviation variables are calculated for each case from the 10 previously created cases. The first 10 cases are deleted from the dataset to allow for these calculations.

Lastly, the Created_on_Weekend variable was created as a binary entry to determine if a case was created on a weekend day (Saturday or Sunday).

Figure 24 shows boxplots for Time Taken for cases either created on a weekend or on a weekday. The mean Time taken for each is shown as a yellow dashed line. As shown previously, very few cases are created on weekends however as can be seen in this figure, cases which are created over the weekend are processed faster on average. The interquartile range is also smaller for cases created on weekends.

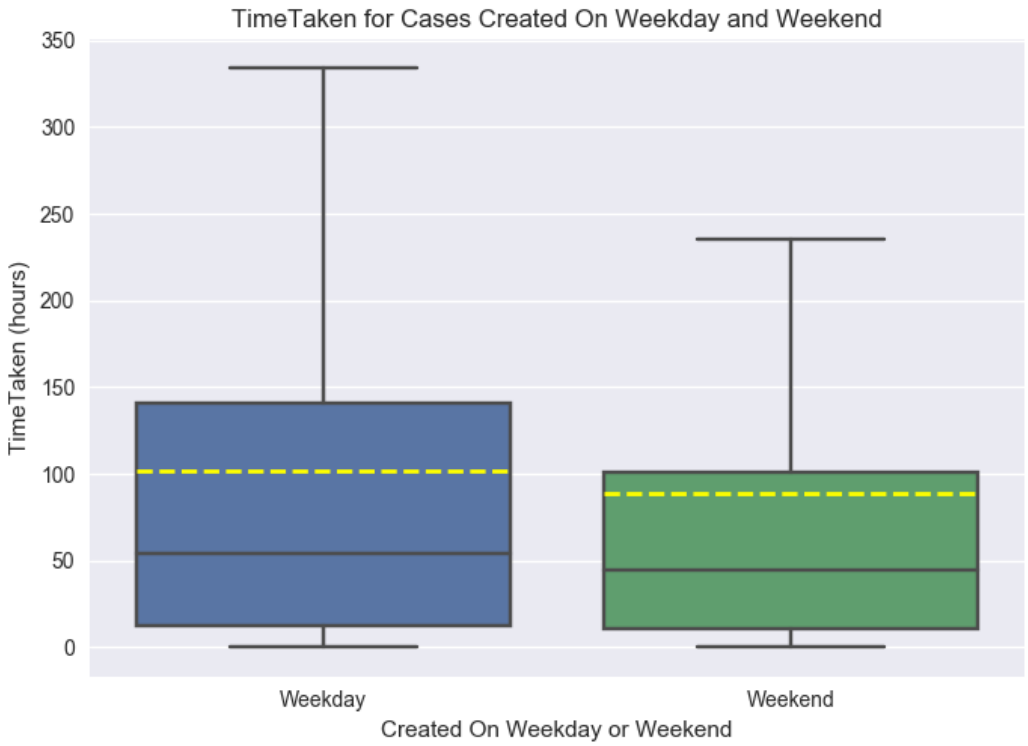

Figure 24: Time Taken Created_on_Weekend boxplot

5.6 Final Dataset

Many variables in the raw dataset were redundant, mostly null or 0 entries or were indicated as meaningless in the context of the objectives given. The full list of variables kept and removed is included in Appendix 3. The resulting pre-processed dataset is summarised in Table 2.

Table 2: Variables in final dataset

Variable	Binary	Numeric	One hot	Ordinal	Total
Original (unaltered)	4	1	0	0	5
Transformed	0	0	10	3	13
Imported	0	2	2	0	4
Generated	1	10	0	0	11
Total	5	13	12	3	33

Chapter 6 - Modelling Techniques

Unless stated otherwise, each modelling technique was implemented using the Sci-kit Learn (Pedregosa et al., 2017) package for Python and the default parameters for each algorithm were used.

6.1 Linear Regression

Linear Regression (LR) is a method of linearly mapping independent input variables to a dependent output variable. The Sci-kit Learn implementation of Linear Regression uses a least squares method.

Linear Regression is one of the most basic machine learning algorithms and so was included in The Practicum as a baseline comparison.

Algorithm and parameters used:

- sklearn.linear_model – LinearRegression

6.2 Kernel Ridge Regression

Ridge Regression is a regularization technique to minimise the effects of collinearity by introducing biases to the low absolute values of the Linear Regression coefficients. The Sklearn Kernel Ridge Regression implementation uses Ridge Regression along with the kernel method to model relationships.

A Kernel Ridge Regression model was implemented in The Practicum due to its strong multicollinearity capability however it was found to produce poor results. The model was overly sensitive to imperfectly-conditioned inputs, required significant computational resources and did not have good accuracy when compared to other algorithms tested. Because of this, Kernel Ridge Regression was omitted from Chapter 9: Results & Discussion.

Algorithm and parameters used:

- Sklearn.kernel_ridge - KernelRidge
- alpha = 0.1 (Used to improve input conditioning to reduce variance of estimated)

6.3 Elastic Net

Elastic Net (EN) is a regularised linear regression technique which is a combination of L1 Lasso Regression and L2 Ridge Regression methods.

Tibshirani (1996) introduced the Lasso Regression model as a new linear least squares estimator method. Both Ridge and Lasso Regression are implemented as a penalty on the regression coefficients which reduce coefficients towards zero relative to the maximum likelihood estimates (Goeman et al., 2016). The reason for this is to combat overfitting caused by high collinearity or dimensionality. In Lasso the linear transformation coefficients are Laplacian distributed meaning that many of the coefficients are likely to be reduced to exactly zero and fewer coefficients will be reduced by relatively little (Goeman et al., 2016). This therefore functions as a means of variable selection. With Ridge Regression, the coefficients are normally distributed which likely results in all the regression coefficients being small but non-zero (Goeman et al., 2016).

Zou and Hastie (2005) recorded Lasso's limitations as: only selecting at most the number of observations as predictors, selecting only one predictor from a group of highly correlated variables and in certain cases, underperforming compared to Kernel Ridge. Zou and Hastie (2005) created Elastic Net which aimed to build on the Lasso method by addressing these limitations. By combining these two penalties, Elastic Net tends to result in more penalisation for all regression coefficients but fewer coefficients reduced to zero than for just Lasso. Therefore, the premise of the Elastic Net is to create a sparse model like Lasso, but also allow the inclusion of correlated predictor groups.

Elastic Net was selected to model the dataset due the large number of variables available. The Elastic Net algorithm will select the most important predictors and disregard variables which were not useful.

Algorithm and parameters used:

- sklearn.linear_model - ElasticNet
- alpha = 100 (Constant to multiply penalty coefficients)
- l1_ratio = 1 (Mixing parameter for L1 and L2 where l1_ratio = 1 is the lasso penalty only)

- max_iter = 100000 (Maximum permissible iterations)

6.4 Random Forest Regression

Decision Trees are a machine learning approach used to discretely categorize data by forming a structure of branches and leaves, where the branches represent logical junctions based on predictors and the leaves represent class labels. Random Forests build on this by generating numerous Decision Trees and then outputting the mode output of those individual trees. Random Forest is among the best performing classifiers and is particularly strong at minimising overfitting (Breiman, 2001). While Random Forests are traditionally used for classification, Liaw and Wiener (2002) showed that random forests are also very effective when applied to regression problems. Random Forest Regression (RFR) works similarly to the classification version of the algorithm, but outputs the mean prediction of the individual trees instead of the mode.

Random Forest Regression was used in The Practicum due its non-linear properties.

Algorithm and parameters used:

- sklearn.ensemble - RandomForestRegressor
- n_estimators = "n_estimators" (Number of trees in the forest. "N_estimators" = 100 for experiments)
- random_state = "seed" (Seed for random number generator. "seed" = 1234 for experimentations)
- max_depth = 25 (Maximum depth of tree)
- n_jobs = -1 (Number of jobs to run in parallel. -1 corresponds to machine cores)

6.5 Gradient Boosting Regression

Gradient Boosting Regression (GBR) is a regression algorithm implementation of the Gradient Tree Boosting model proposed by Friedman (2001). Gradient Boosting Regression, like Random Forest Regression, is a tree ensemble model which sums together the predictions of multiple weak (classification and regression trees (CART)) prediction models. Boosting algorithms add new trees sequentially, without altering existing trees, to correct the errors made by the existing trees until no further improvement can be made or a fixed number of trees has been reached. Gradient Boosting uses the gradient descent algorithm to optimise parameters in order to minimize the residual loss function when adding new trees. Thus, gradient descent is

not used to optimize parameters, but instead is used to optimize the residual loss when a new tree is added to the model, i.e. the new tree is parameterised and the parameters of the tree are altered so that the residual loss is minimized. So, a cost function is optimised by gradient descent in function space through an iterative process which chooses a function that points in the negative gradient direction.

Algorithm and parameters used:

- sklearn.ensemble - GradientBoostingRegressor
- random_state = "seed" (Seed for random number generator. "seed" = 1234 for experimentations)

6.6 Grid Search

Grid Search was used to optimise the hyperparameters for each of the algorithms used. For each algorithm, a large selection of hyperparameters was fed into the Grid Search algorithm and the optimal hyperparameters were returned. The hyperparameter combinations which resulted in the highest R squared value were judged to be optimal for each algorithm. These hyperparameters were then used in all further experiments.

Chapter 7 - Evaluation Techniques

7.1 Cross Validation

For each model, the input dataset was split into train and test data using k-fold Cross Validation with the number of folds, k, taken as an input parameter. Cross Validation splits the original input sample randomly into k equal sized subsamples of data. One of the subsamples is retained for testing the model's accuracy and k-1 subsamples are used to train the model. This process is repeated k times with each of the k subsamples used once to test the model. The k results are then averaged to give a single estimated prediction for the entire sample.

The advantage of using this technique is that each of the observations is used for both testing and training, but is only used for testing purposes once. Cross Validation was used instead of a simpler train/test split method to obtain more robust prediction results while minimising overfitting.

After the model was trained on each subsample of data, a set of training metrics was returned which described various predictive errors that resulted from the model predicting the Time Taken for the data it was just trained on, i.e. training metrics refer to the averaged errors produced when the model is used to predict the Time Taken for data it has already been trained on at each fold.

Test metrics refer to the averaged errors produced when the model is used to predict the Time Taken for each subsample not used to train it at each fold.

The standard deviation was also returned for both the training and testing stages.

In Chapter 9: Results & Discussion, unless stated otherwise, only the results from the testing stages of each experiment are shown.

7.2 R squared

R squared, or the coefficient of determination, is the first measure implemented in The Practicum to examine the difference between the predicted and actual values. R squared assesses the model's ability to make predictions by showing the proportion of the variance in the dependent variable, y, that is predicted from the independent variable(s), X. The best

possible value for this metric is 1, which would indicate that all the variation in y is accounted for by X. A value of 0 would indicate that X accounts for none of the variation in y. A value of 0.75 would indicate 75% of the variation in y is explained by X. The formula is as follows:

$$SS_{res} = \sum_{i=1}^{n}(y_i - f_i)^2 = \sum_{i=1}^{n} e_i^2$$

$$SS_{tot} = \sum_{i=1}^{n}(y_i - \bar{y})^2$$

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

SSres is the residual sum of squares, this sums the squares of each actual value minus its predicted value. SStot is the total sum of squares, this is the sum of the squared differences between each observation and the mean.

7.3 Root Mean Square Error

Root mean square error (RMSE) is the second measure used to examine the difference between the predicted and actual values. For this metric, a smaller value is better because the larger the RMSE, the larger the errors in the predictions made. The RMSE is in the same units used by the model which means that if the model was predicting the Time Taken to do something in hours, the RMSE would also be in hours. The formula is as follows:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n}}$$

7.4 Mean Absolute Error

The Mean Absolute Error (MeanAE), is the mean of the absolute differences between the actual and predicted values. For this metric, again a smaller value is better. The formula is as follows:

$$MeanAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

The MeanAE is always equal to or less than the RMSE, and the greater the difference between the RMSE and MeanAE, the greater the variance in the individual errors in the sample.

Unlike with the RMSE, the MeanAE doesn't square the errors before taking the mean and therefore, unlike RMSE, it does not give a relatively larger weight to larger errors. The RMSE is a more useful metric when larger errors are particularly undesirable because it penalises larger errors more. However, comparing the RMSE from different sample sizes can be a problem because the RMSE can tend to be larger as the sample size increases.

7.5 Median Absolute Error

The Median Absolute Error (MedianAE), or median absolute deviation, is the median of the absolute differences between the actual and predicted values. Again, smaller values are better for this metric. The formula is as follows:

$$MedianAE = median(|y_i - median(Y)|)$$

This metric has the advantage of being insensitive to outliers because it takes the median of the absolute values of the residuals and the median is relatively unaffected by values at the tails. In comparison, the RMSE is highly sensitive to outliers and the MeanAE is slightly sensitive to outliers.

7.6 Number of Correct Predictions

The fifth and final metric used to investigate prediction results was the percentage of correct predictions within a given margin of error. This was calculated as a part of the program's output in accordance to the business's needs. These were the percentage of correct predictions within +/- 1 hour (the lowest amount of granularity required), +/- 4 hours (a full working day), +/- 8 hours (a double shift used on the last working day of the month), +/- 12 hours (a full day), +/- 24 hours (two full days), +/- 48 hours (4 working days, the guaranteed cut off period before the end of each month that Microsoft is most interested in predicting processing times for).

7.7 Predicted versus Actual Time Taken Plots

A visual indication of the accuracy of predictions was included in the form of scatterplots. The actual Time Taken is shown on the x-axis and the Time Taken predicted by the model used is shown on the y-axis. Each data point on the plot represents a prediction made by the model. For

these plots a good result is a neat diagonal shaped plot which means the predicted values are not far away from their actual values.

7.8 Importances

Importances for each tree based algorithm (RFR and Gradient Boosting) were printed and saved to show the influence of each variable. These importances sum to 1, with a larger value indicating that the variable was more important to the model's predictions.

For the Linear Regression and Elastic Net algorithms, the coefficients returned by each algorithm were scaled according to the distribution of each input column to return a list of variables ordered by how much they each influenced the algorithms. This was done using Sklearn's StandardScaler which standardises each coefficient by subtracting the mean of the corresponding column and dividing by its standard deviation. This shows the mean change in the response given a one standard deviation change in the predictor. These importances do not sum to 1 but are instead a (scaled) positive or negative representation of how important each coefficient was to the algorithm so that if one coefficient is larger in magnitude than the others, it was more important to the algorithm. A second percentage importance value was also calculated using these standardised importances as an input. This percentage importance value sums to 1.

Chapter 8 - Experimentation & Case Study

This chapter describes experiments carried out to evaluate the effectiveness of each preprocessing and modelling technique described in Chapter 5: Preprocessing Techniques and Chapter 6: Modelling Techniques. Each experiment included a baseline calculation followed by the addition of increasingly complex preprocessing techniques which added variables to the prepared dataset. Each experiment used the variables from the previous experiment plus additional variables. Lastly, case studies were conducted using the final model to simulate scenarios that Microsoft will face.

Both the Time Taken variable generated for each experiment and the filtering stage of the program remained constant throughout the experimentation for consistency. All experimentation included the error metrics and plotting described previously and model outputs are recorded Chapter 9: Results & Discussion.

8.1 A. Baseline

A baseline of the mean Time Taken for all resolved cases was calculated. While trivial, this average value is what the Microsoft BPO team currently use to estimate the workload for pending cases and so is a valid comparison for the proposed problem solution.

By comparing the baseline Time Taken value against the actual Time Taken values for each case, the coefficient of determination, the root mean squared error, the mean absolute error and the median absolute error were calculated.

8.2 B. Variable Experiments

8.2.1 B1. Readily Available Data

The first experiment used readily available data that was already in a useable format for modelling. This was done using only the variables listed below which were all in the form of numeric or binary entries.

1. IsGovernment
2. AmountinUSD
3. IsMagnumCase
4. IsSignature
5. IsSOXCase

8.2.2 B2. Transformed Features

The second experiment added variables that were readily available but not in a usable form. This was achieved by including ordinal variable mapping and one-hot encoding preprocessing steps. The ordinally mapped variables were as follows:

1. Priority
2. Complexity
3. StageName

The one-hot encoded variables were as follows:

1. CountrySource
2. CountryProcessed
3. SalesLocation
4. StatusReason
5. SubReason
6. ROCName
7. Sourcesystem
8. Source
9. Revenutype
10. Queue

8.2.3 B3. Imported Data

The third experiment was to include the data mined from the vw_HoldActivity activity and vw_AuditHistory worksheets as described in the preprocessing chapter.

8.2.4 B4. Generated Variables

The fourth experiment included all the variables that were generated and so not originally in the available dataset. These variables were constructed from the existing Created_On and ResolvedDate variables along with the generated Time Taken variable and were as follows:

1. Concurrent_open_cases
2. Cases_created_within_past_8_hours
3. Cases_resolved_within_past_8_hours
4. Seconds_left_Day
5. Seconds_left_Month
6. Seconds_left_Qtr
7. Created_on_Weekend

8. Rolling_Mean
9. Rolling_Median
10. Rolling_Std

8.2.5 B5. Seconds to End of Year

In addition to all the variables up to this point, the seconds to end of year was included in a side experiment. The reason this was not included as a core variable is due to the limited dataset available (only ~6 months of data from February to August was available for The Practicum).

8.2.6 Mandatory Variables

When a case is created initially there are several required variables which must be to be filled in before the case can be added to Microsoft's case logging system. This experiment was created to predict the time taken to process cases when only the mandatory fields are filled in. Of these mandatory variables, only the following 5 variables are not eliminated by the preprocessing program:

1. CountryProcessed
2. CountrySource
3. Queue
4. ROCName
5. SalesLocation

In addition to these 5 variables, 6 other variables are included in this experiment because they can be constructed using only the Created_On timestamp for each case. These are listed as follows:

1. Cases_created_within_past_8_hours
2. Concurrent_open_cases
3. Created_on_Weekend
4. Seconds_left_Day
5. Seconds_left_Month
6. Seconds_left_Qtr

8.2.7 Minimum Variables

A model was created which uses only the 6 input variables which could be constructed using the timestamps assigned to each case when it is created. These are listed as follows:

1. Cases_created_within_past_8_hours
2. Concurrent_open_cases
3. Created_on_Weekend
4. Seconds_left_Day
5. Seconds_left_Month
6. Seconds_left_Qtr

This experiment simulates a situation where Microsoft want to know the time it will take to process cases when they have estimated the number of orders they will receive in the future and when they will receive them.

8.3 Simulated Case Studies

8.3.1 Experiment C: Testing with Unseen Data from Specific Periods

This experiment was carried out to determine each model's performance when they were used to predict the Time Taken for cases from a specific time period which was not used to train them, i.e. the models were trained using cases from one period of time and tested using cases from another.

This experiment consisted of two stages: a training stage and a testing stage. The data was split into two subsets, one for each stage. In the training stage, the models were trained on one subset of data, and in the testing stage the same models were used to predict the Time Taken values for the second unseen subset of data.

Two different sub experiments were carried out using two different splits for training and testing.

8.3.2 Experiment C1: Testing with Unseen June Data

The training stage used data collected pre-June and the testing stage used data collected in June only.

8.3.3 Experiment C2: Testing with Unseen July Data

The training stage used data collected pre-July and the testing stage used data collected in July only.

In both experiments the following mandatory variables were used:

1. CountryProcessed
2. CountrySource
3. Queue
4. ROCName
5. SalesLocation

The following variables which can be constructed from each case's Created_On timestamp were also used:

1. Cases_created_within_past_8_hours
2. Concurrent_open_cases
3. Created_on_Weekend
4. Seconds_left_Day
5. Seconds_left_Month
6. Seconds_left_Qtr

Unlike in Cross Validation, where over k folds the model is trained with 90% of the data, tested on the unseen 10% and results are averaged, in this experiment the model is trained only once using pre-June or pre-July data and tested only once using June or July data respectively. Using Cross Validation, it is extremely likely that cases from each month are used to both train and test the model at each fold. Therefore, each month's variable's distributions will have been used to train the model. This means that extrapolating whether the model can accurately predict for unseen months would not work.

Discovering whether the model, using the data available now, can accurately make predictions for data collected later is important to know because a full year of data was not available for The Practicum. If the results of this experiment are poor but other experiments have good results, a recommendation can be made to Microsoft to wait until a full year of data has been collected and used to train the model before the model is considered for implementation

GitHub link: https://github.com/K-Ellis/Predicting-Transaction-Times

Chapter 9 - Results & Discussion

9.1 Experimentation Results

9.1.1 Main Results

As more variables were included in the model, the accuracy metrics improved significantly. The percentage correct within an error margin of +/- 48 hours (4 days) and RMSE after variables were included at each stage of experimentation are shown in Figure 25 and Figure 26 respectively for each algorithm.

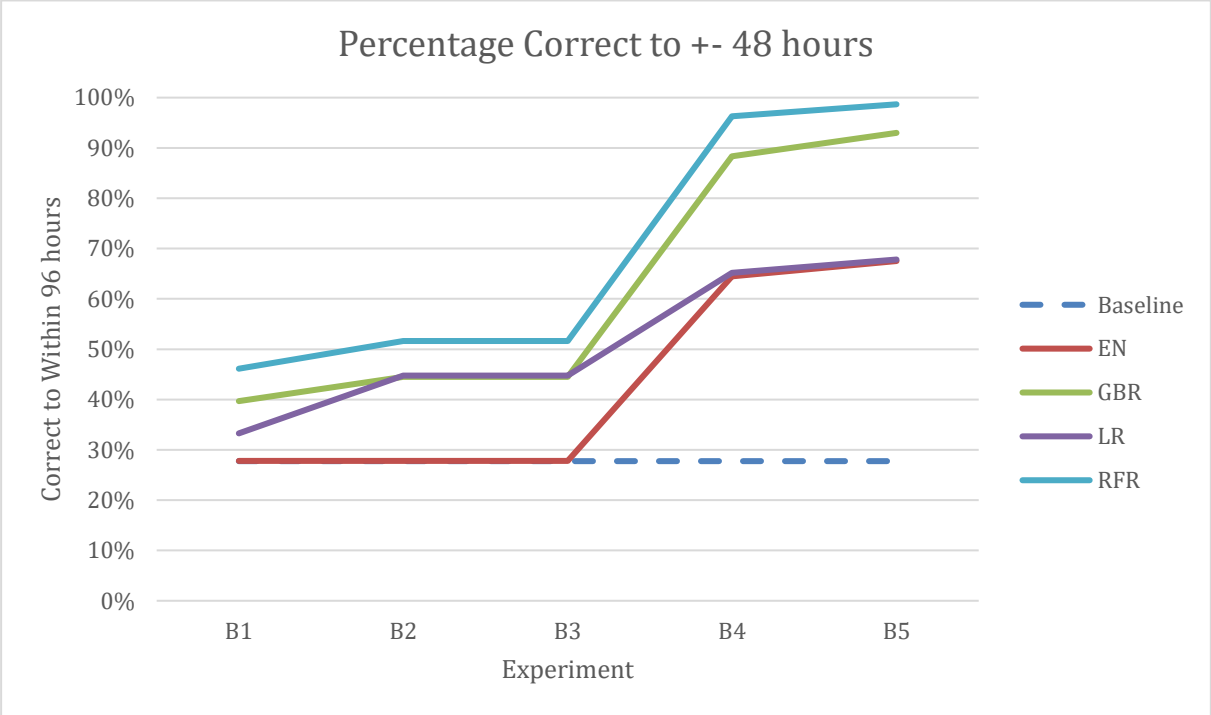

Figure 25: Correct predictions within +/- 48hrs error margin for each experiment

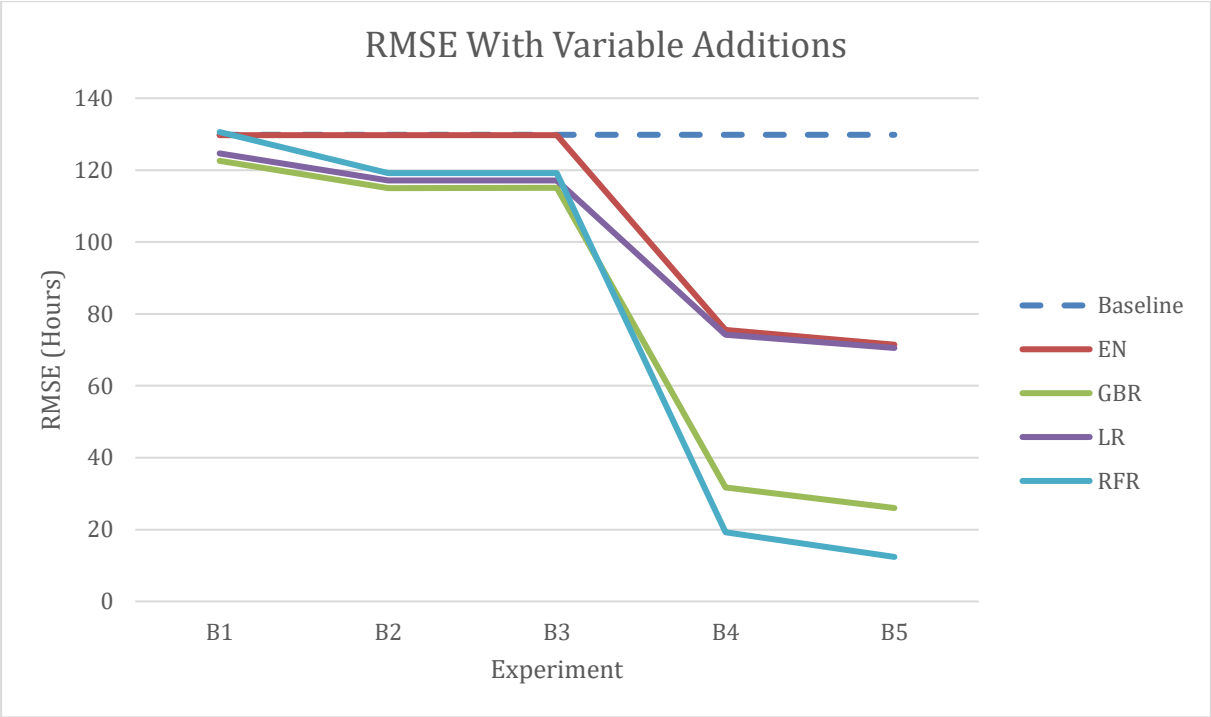

Figure 26: RMSE comparison of experiments

Experiment B4 included all variables except for the seconds to end of year variable. Since this experiment represents the most complete model, it has been examined in more detail. The percentage correct predictions within an error margin of +/- given times is shown in Figure 27 and we can see increasing accuracy over larger time frames. All algorithms have outperformed the baseline and RFR gave by far the best results, reaching over 90% correct predictions within an error margin of +/- 24 hours.

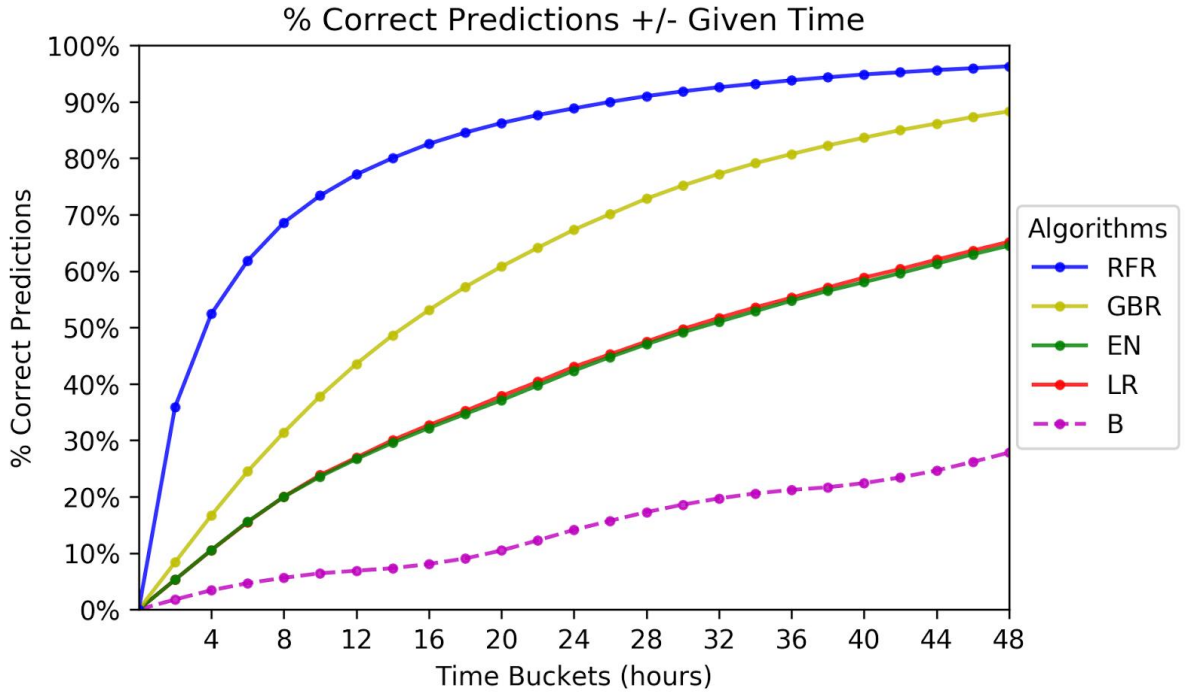

Figure 27: Most predictive model correct predictions over time algorithm comparison

Table 3 records the error metrics for each algorithm in experiment B4. The results are mean values from 10-fold cross validation testing and units are in hours for RMSE, MeanAE and MedianAE. As can be seen, RFR gave the best results for each metric and the Baseline was outperformed by all the machine learning algorithms for each metric. LR and EN gave similar results indicating that EN's regularisation had little effect on its predictions compared to LR's predictions.

Table 3: Most predictive model error metrics

Metric	Baseline	LR (+/-)	EN (+/-)	RFR (+/-)	GBR (+/-)
R2	0	0.67 (0.01)	0.66 (0.01)	0.98 (0.00)	0.94 (0.00)
RMSE	129.46	74.24 (2.22)	75.56 (2.12)	19.21 (0.93)	31.69 (0.89)
MeanAE	90.82	47.23 (0.99)	48.17 (1.04)	9.80 (0.42)	22.03 (0.54)
MedianAE	76.54	30.28 (1.12)	30.92 (1.22)	3.63 (0.21)	14.55 (0.39)
+/- 48 hrs	27.82%	65.19%	64.50%	96.30%	88.31%

Plots of the predicted versus actual Time Taken values for the Linear Regression, Elastic Net, Random Forest Regression and Gradient Boosting Regression algorithms are shown in Figure 28, Figure 29, Figure 30 and Figure 31. A diagonal clustering of points on these plots represents

strong predictions and a best fit line for the data points of the plots has been included for reference. As can be seen, both linear models performed similarly and tended to under predict Time Taken values as the actual value got larger. GBR performed better than both linear models, but RFR clearly has be best plot which is a neat diagonal line with far fewer outliers than GBR.

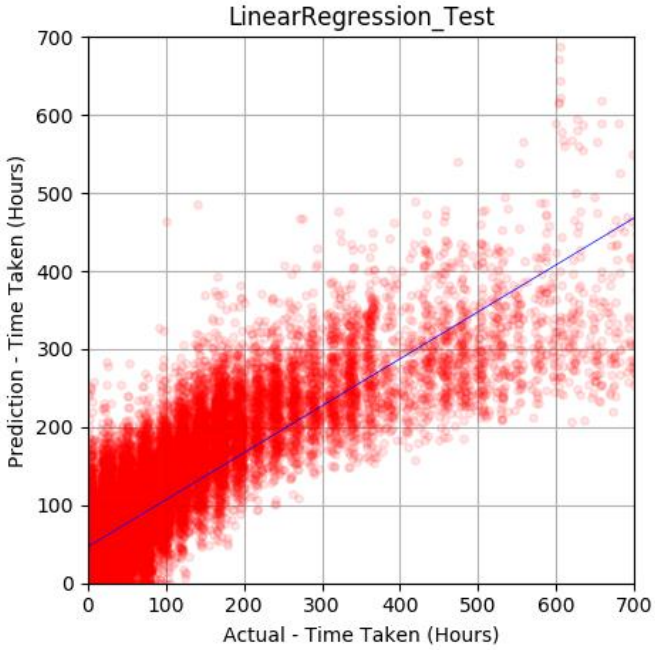

Figure 28: Experiment B4 test data - actual versus predicted time taken (LR)

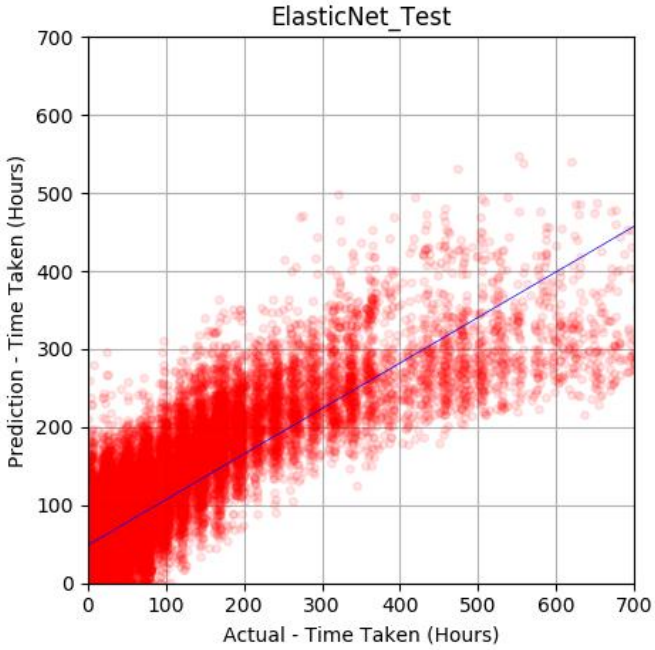

Figure 29: Experiment B4 test data - actual versus predicted time taken (EN)

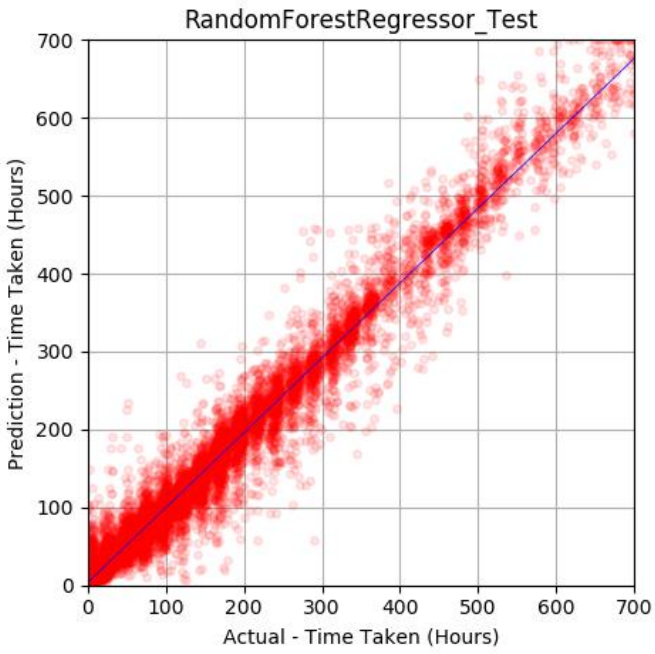

Figure 30: Experiment B4 test data - actual versus predicted time taken (RFR)

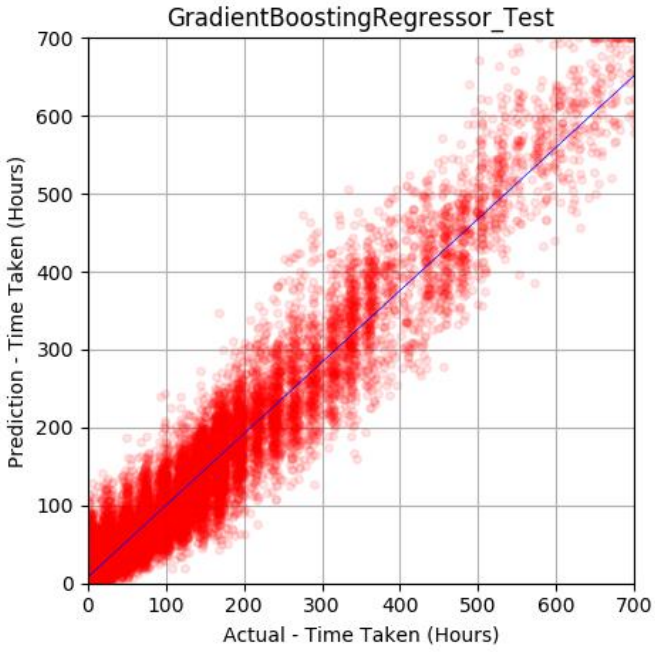

Figure 31: Experiment B4 test data - actual versus predicted time taken (GBR)

9.1.2 Importances Table for Algorithms

The importances assigned to the top variables as identified by each algorithm is shown in Table 4. As can be seen, the two tree based algorithms placed most importance on the number of concurrently open cases and other extrinsic/constructed variables. Both linear algorithms placed

a lot of importance on intrinsic variables recorded directly by Microsoft's CRM case logging system such as a case's StatusReason and Priority.

Table 4: Most predictive model algorithm importances

	LR	EN	RFR	GBR
1	StatusReason_Ready for Audit	sourcesystem_CLT	Concurrent_open_cases	Concurrent_open_cases
2	StatusReason_Customer Hold	Priority	Rolling_Std	Seconds_left_Qtr
3	sourcesystem_NEMEC	Complexity	Rolling_Mean	Seconds_left_Month
4	Queue_Broken	StageName	Seconds_left_Qtr	Rolling_Mean
5	sourcesystem_`	Cases_resolved_within_past_8_hours	Seconds_left_Month	Rolling_Std
6	sourcesystem_web	Source_Web	Rolling_Median	Rolling_Median
7	sourcesystem_clt	Cases_created_within_past_8_hours	Cases_resolved_within_past_8_hours	Cases_created_within_past_8_hours
8	StatusReason_Ready for Archiving	Concurrent_open_cases	Seconds_left_Day	Cases_resolved_within_past_8_hours
9	sourcesystem_Web	Rolling_Std	Cases_created_within_past_8_hours	Seconds_left_Day
10	sourcesystem_Aplquest	Rolling_Mean	SubReason_Basic Enterprise Commitment	CountryProcessed_northamerica

9.1.3 RMSE Over Time

The RMSE values over time are examined to determine which periods of time had the largest predictive errors. RFR is included because this algorithm gave the best predictive results as shown previously and LR is included as a baseline to compare against.

Figure 32 and Figure 33 show the RMSE for the time of the day that cases were created for LR and RFR. For both algorithms, the RMSE being largest later in the day means that there is more error when predicting the Time Taken for cases created during this time. When a case is created outside of these times, there is more certainty in predicting the case's Time Taken. This could be because when a case is created earlier in the day, Microsoft's agents are likely to process it before the end of the day, but when a case is created later in the day it might not be resolved before they finish work, and so the case will be left idle while the agent is not working, making it much harder to predict when it will be resolved. If the case was created preceding the weekend or a public holiday, it might be many days before the agent can continue processing the case.

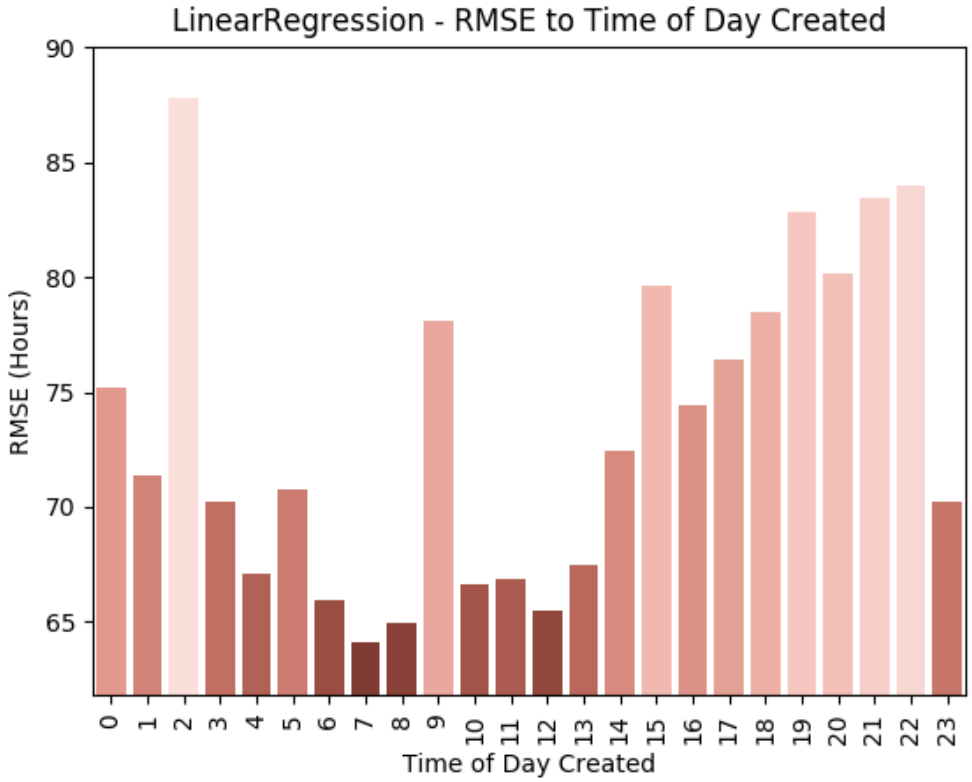

Figure 32: Experiment B4 RMSE to time of day (LR)

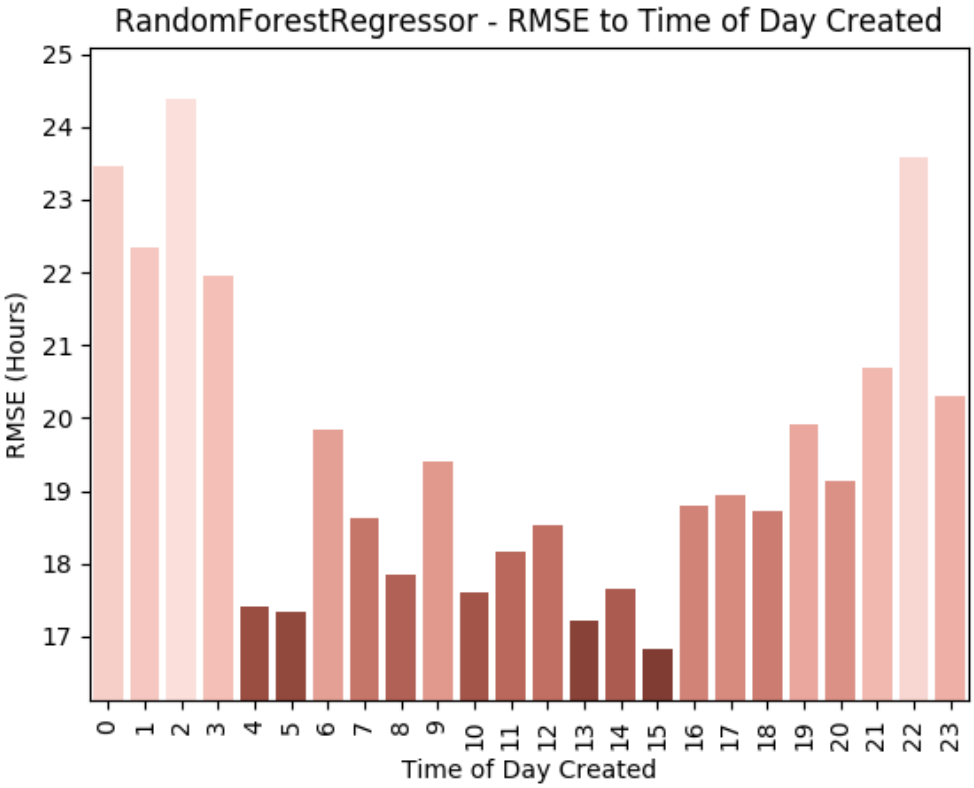

Figure 33: Experiment B4 RMSE to time of day (RFR)

However, as can be seen in Figures 34 and 35, it is not cases created on a Friday which have the largest RMSE, but rather for LR it is cases which were created on a Wednesday and for RFR it is cases created on weekends. Because the average time to process a case is approximately 4 days, this suggests that cases created on a Wednesday are often not finished before the weekend, and so often extend past 4 days before they are resolved which would make it much harder to predict the Time Taken for these cases. RFR struggling with predicting cases created on weekends could be caused by the smaller volume of cases created over weekends, i.e. the smaller sample size could be causing the larger errors for these days.

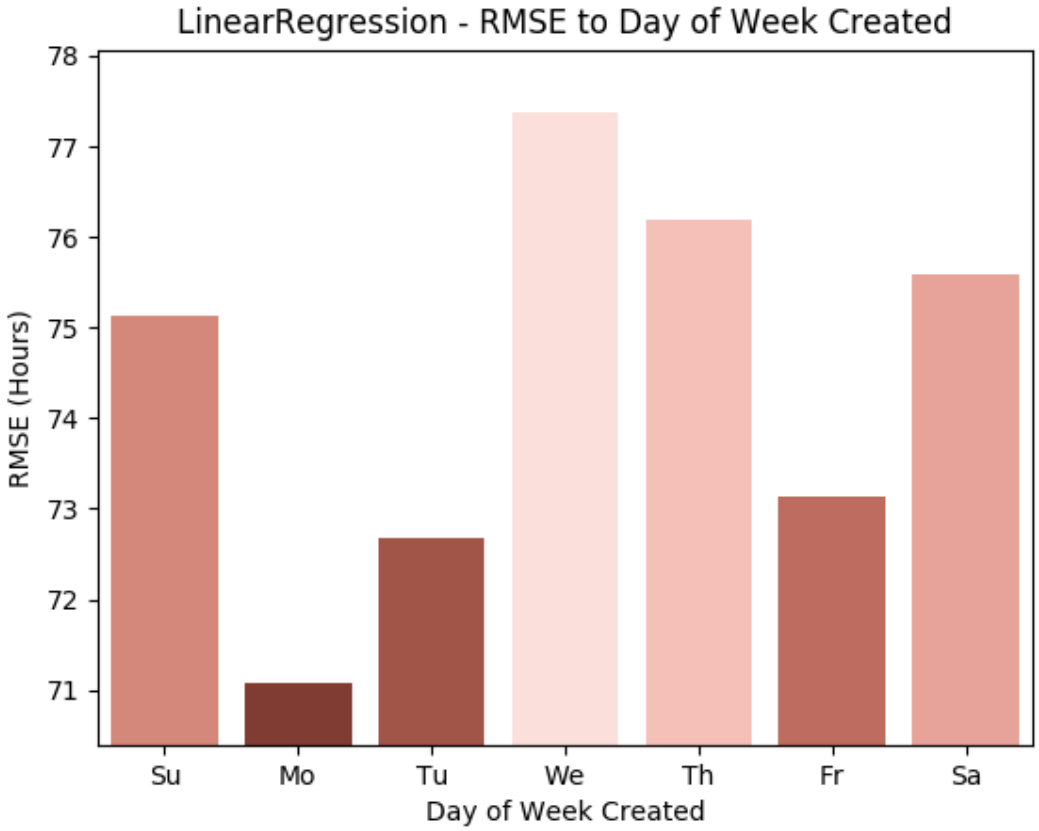

Figure 34: Experiment B4 RMSE to day of week (LR)

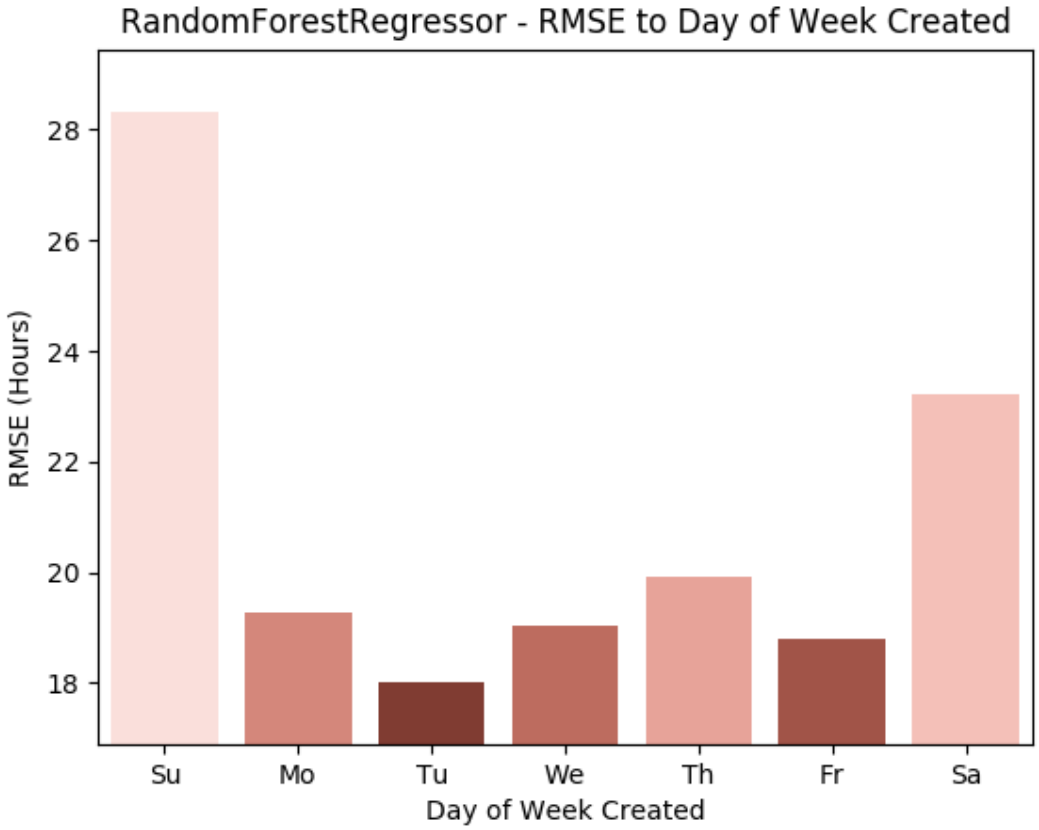

Figure 35: Experiment B4 RMSE to day of week (RFR)

Figures 36 and 37 show the RMSE for each day of the month a case was created on for LR and RFR. The RMSE values for LR are smallest midway through the month and then increase steadily towards the end of each month before being largest directly preceding the end of each month. For RFR, the RMSE was large for the first half of the month and small for the second half of each month.

A smaller volume of cases comes into Microsoft at the start of each month. This smaller sample size could be causing the larger RMSE values here. It is also possible that the larger RMSE values at the start of the month could be due to Microsoft's agents working at a less consistent pace at the start of each month. During the first half of each month agents are likely to be more relaxed about end of month deadlines causing their work rates to be inconsistent, therefore predicting the Time Taken to resolve a case created during this time is likely to be much harder.

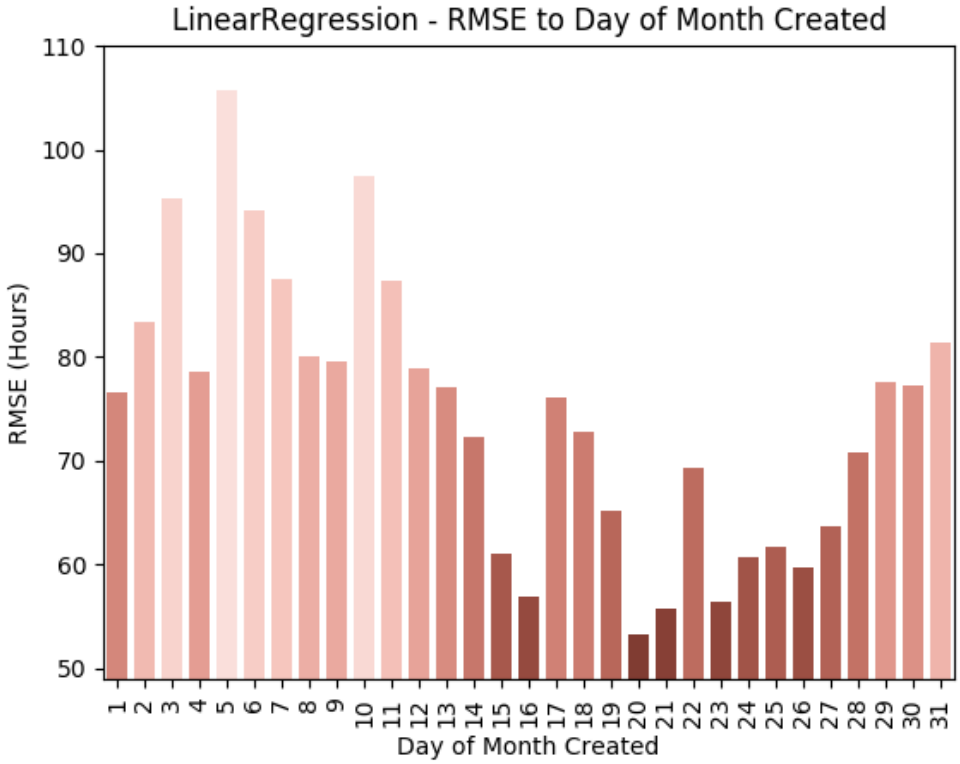

Figure 36: Experiment B4 RMSE to day of month (LR)

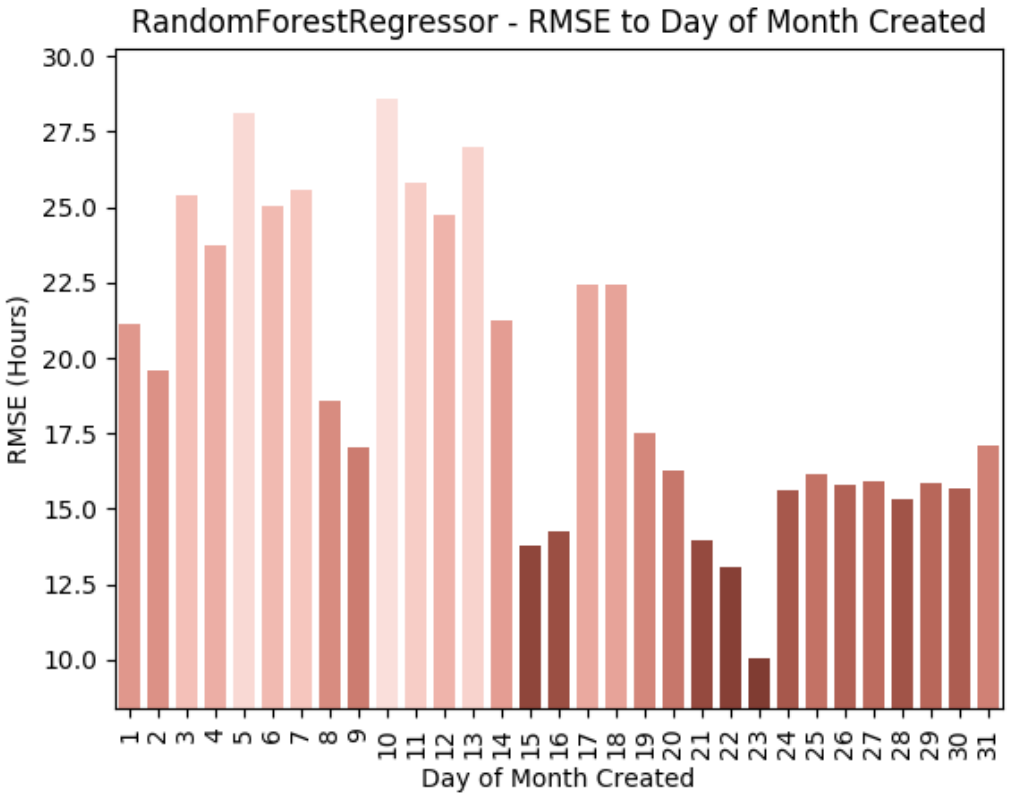

Figure 37: Experiment B4 RMSE to day of month (RFR)

As can be seen in Figures 38 and 39, the RMSE values for both algorithms were largest at the start of each quarter. It can clearly be seen that the RFR has less error predicting the Time Taken for cases created in the second half of each quarter. Again, a smaller volume of cases comes into Microsoft at the start of each quarter which could be causing the larger RMSE values. Also, although the end of month deadlines put a lot of pressure on Microsoft's agents to process cases quickly, the end of quarter deadlines are even more important to the company. This could mean that these cases are much more predictable compared to cases created at the start of the quarter, when agents are less worried about end of quarter deadlines. Thus, case processing times are likely to vary much more and be harder to predict at the start of each quarter.

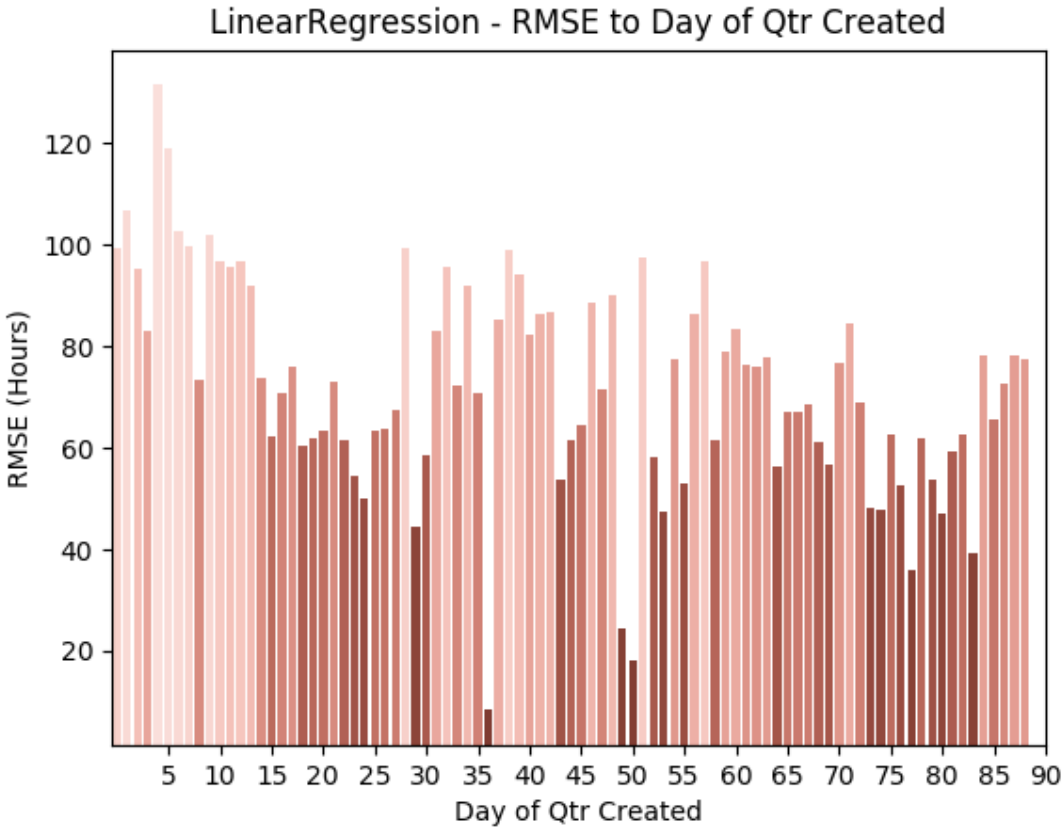

Figure 38: Experiment B4 RMSE to day of quarter (LR)

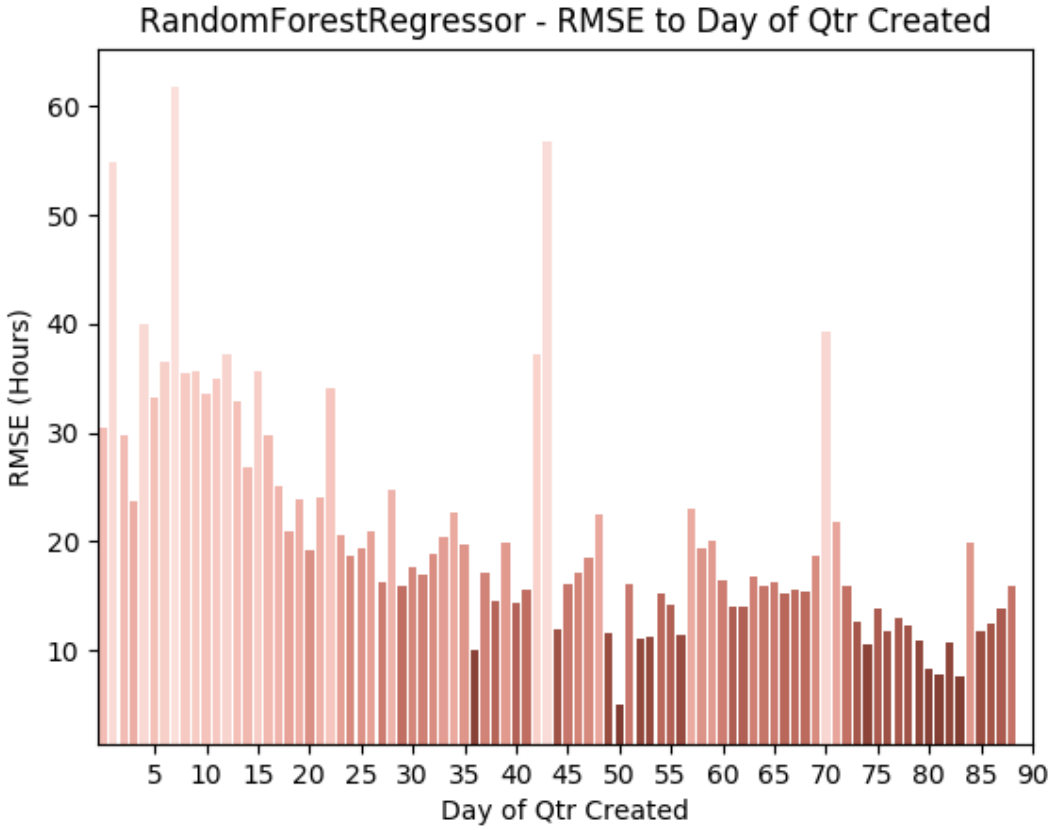

Figure 39: Experiment B4 RMSE to day of quarter (RFR)

9.1.4 Mandatory and Minimum Variables

When the models were trained using only the mandatory variables which must be filled in for a case to be logged by Microsoft's CRM system, the testing results for EN, LR and GBR were only slightly worse than when all the variables were used in the main B4 experiment and the RFR results were nearly as good.

Once again, as seen in Table 5, LR and EN had very similar metrics and each of the machine learning algorithms outperformed the baseline for each metric measured. RFR continued its dominance over the other algorithms with very impressive results despite the few variables used in this experiment.

Table 5: Mandatory variables error metrics

Metric	**Baseline**	**LR (+/-)**	**EN (+/-)**	**RFR (+/-)**	**GBR (+/-)**
R2	0	0.54 (0.01)	0.53 (0.01)	0.97 (0.01)	0.90 (0.01)
RMSE	129.46	87.73 (2.69)	88.67 (2.76)	22.36 (2.10)	40.37 (1.07)
MeanAE	90.82	54.51 (1.12)	55.04 (1.14)	8.15 (0.30)	26.45 (0.52)
MedianAE	76.54	33.82 (0.70)	33.13 (0.68)	2.01 (0.05)	17.03 (0.22)
+/- 48 hrs	27.82%	60.67%	60.62%	96.63%	84.92%

LR generally had opposite feature importances to the other 3 algorithms, with LR favouring the intrinsic mandatory features and the others favouring the extrinsic constructed features. These importances can be seen in Appendix 4.

However, when the model is trained using only the minimum variables which can be created using the Created_On timestamp assigned to each case, the testing results are almost identical to those seen in Table 6 when the Mandatory variables were used.

Table 6: Minimum variables error metrics

Metric	**Baseline**	**EN (+/-)**	**LR (+/-)**	**RFR (+/-)**	**GBR (+/-)**
R2	0	0.53 (0.01)	0.54 (0.01)	0.97 (0.00)	0.89 (0.01)
RMSE	129.46	88.33 (2.33)	88.11 (2.31)	22.66 (1.13)	40.95 (0.91)
MeanAE	90.82	54.87 (1.00)	54.86 (0.97)	7.91 (0.35)	26.40 (0.61)
MedianAE	76.54	33.07 (1.15)	33.89 (1.06)	1.89 (0.08)	16.38 (0.57)
% 96hrs	27.82%	60.62%	60.45%	96.68%	84.92%

Figure 40 shows the percentage of correct predictions for each algorithm within an error margin of +/- given times when the Minimum variables are used. The RFR model is very accurate despite the few variables used to train it. It achieves over 90% correct predictions within an error margin of +/- 24 hours whereas the next best algorithm, GBR, does not reach this percentage within +/- 48 hours. At an error margin of +/- 48 hours, RFR reaches ~96% correct predictions, compared to the Baseline which manages just ~28%.

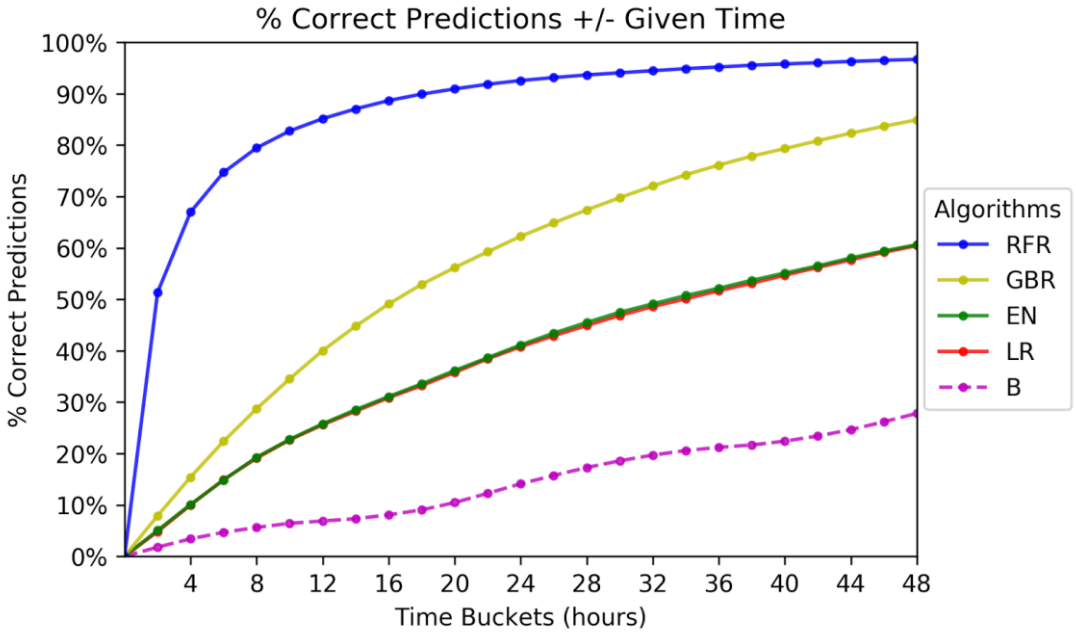

Figure 40: Correct predictions within a given time using minimum variables

Figure 41 shows the RMSE for each algorithm used in the Main B4 experiment, the mandatory variables experiment and the minimum variables experiment. As can be seen, both the mandatory and minimum experiment's RMSE values were basically identical for each algorithm tested. Each algorithm's RMSE was only slightly worse than those from the main B4 experiment and each was much better than when the Baseline was used to predict Time Taken. Considering how few variables were used to train the minimum models, this is an interesting result.

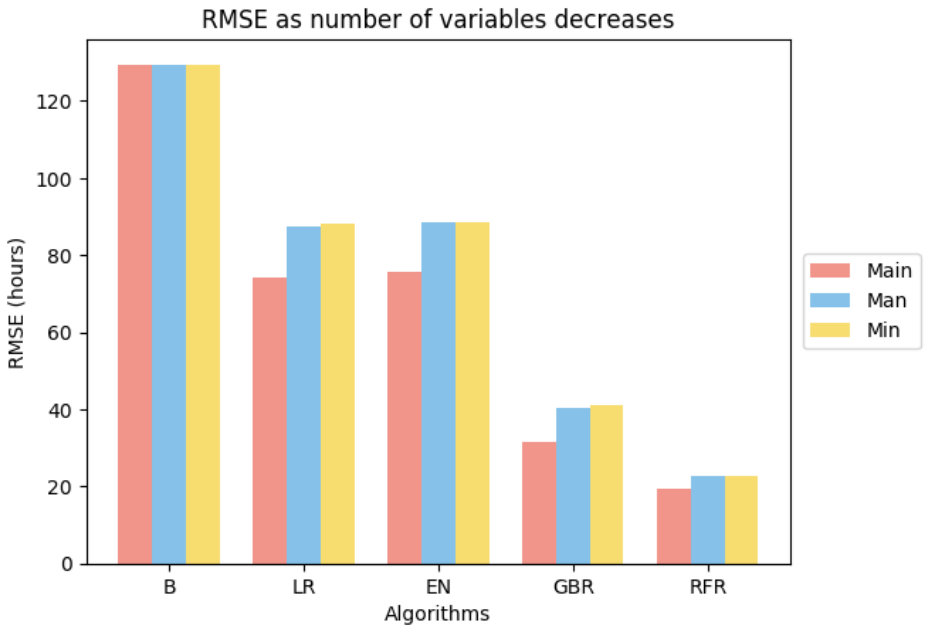

Figure 41: RMSE for algorithms in main, mandatory and minimum experiments

As seen in Figure 42, each algorithm performs only slightly worse when the mandatory and minimum experiment's variables are used compared to the main B4 experiment's variables. However, both the mandatory and minimum experiment's results are nearly identical for each algorithm.

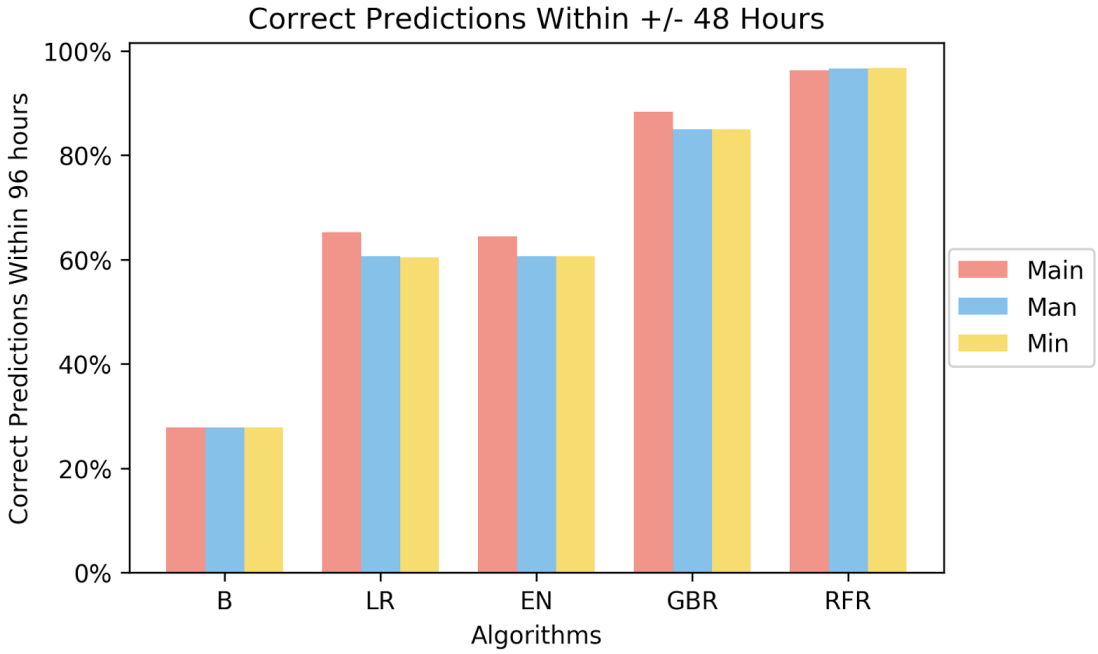

Figure 42: Correct predictions to +/- 48 hours for main, mandatory and minimum

From these results, it can be inferred that the extra variables from the mandatory dataset were not predictive of the Time Taken to process cases. These extra variables, along with many other

variables intrinsic to Microsoft's CRM system, have failed to provide a good basis for prediction, implying that the type of case, where it was processed, its assigned complexity, priority, etc. are not what really dictates the processing time for a case. Instead it is the constructed/extrinsic variables that measure workload and the time until deadlines which are predictive of processing times.

At first glance, this may seem like bad news, i.e. the details of a case having no real impact on its processing times, however, the extrinsic/constructed variables are predictive of processing times. This means that using the minimum Created_On timestamp variables only, accurate predictions can be made about the processing times for a case when the cases has just started processing. This minimum model can be used in an online setting to make predictions in real time for unfinished cases, i.e. models using the minimum variables do not have to wait until a case's fields are filled in before it can make predictions about the time it will take for that case to process.

Figure 43 shows the RMSE for each algorithm when only the minimum Created_On timestamp variables are used. It is clear from this plot and from previous plots that the RFR algorithm performs best.

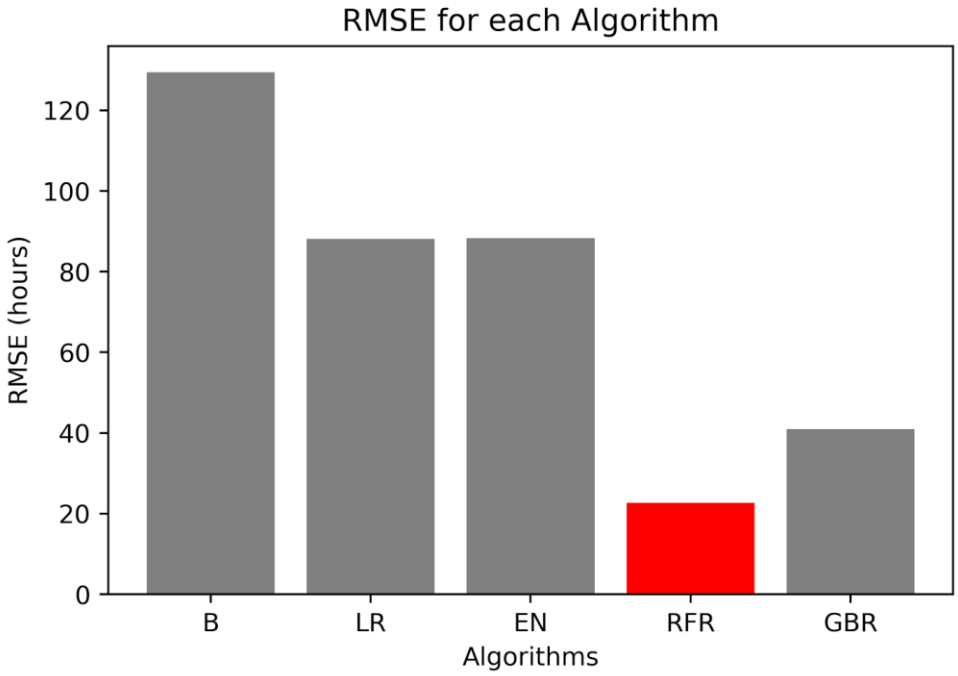

Figure 43: RMSE for each algorithm using minimum variables

Shown in Appendix 5 are the predicted versus actual Time Taken plots and the standardised residual versus predicted Time Taken plots for this experiment. Both linear models struggled when predicting larger values and heteroscedasticity in the predictions can be observed. Each algorithm was outperformed by RFR which displayed good results in each plot.

Appendix 4 shows the variable importances assigned by each algorithm to each variable for this experiment. LR placed most importance on whether a case was created on the weekend. EN shared its importances quite evenly between the number of cases created in the past 8 hours, the number of concurrent open cases and the seconds left until the end of the month and quarter. The two tree based algorithms favoured concurrent open cases most but GBR only placed ~40% of its importance on this variable, ~30% to Seconds_left_Qtr and ~ 19% to Seconds_left_Month, compared to RFR which placed ~78% importance to concurrent_open_cases, ~11% to Seconds_left_Qtr and ~5% to Seconds_left_Month. RFR only placed a total of ~5% of its importance to the other 3 variables, with cases created in the past 8 hours getting ~3%, Seconds_left_Day getting ~1.5% and Created_on_Weekend getting ~0.004%~ importance.

9.2 Simulated Case Study Results

9.2.1 Experiment C1: Testing with Unseen June Data

For this experiment the model was trained on 15,328 pre-June cases and then tested on 8,664 unseen cases from June.

For the training stage of this experiment, each of the algorithms performed in a similar fashion to how they did in previous experiments. Figure 44 shows the familiar story of RFR outperforming the other algorithms, LR and EN having near identical results and all the algorithms doing much better than the baseline.

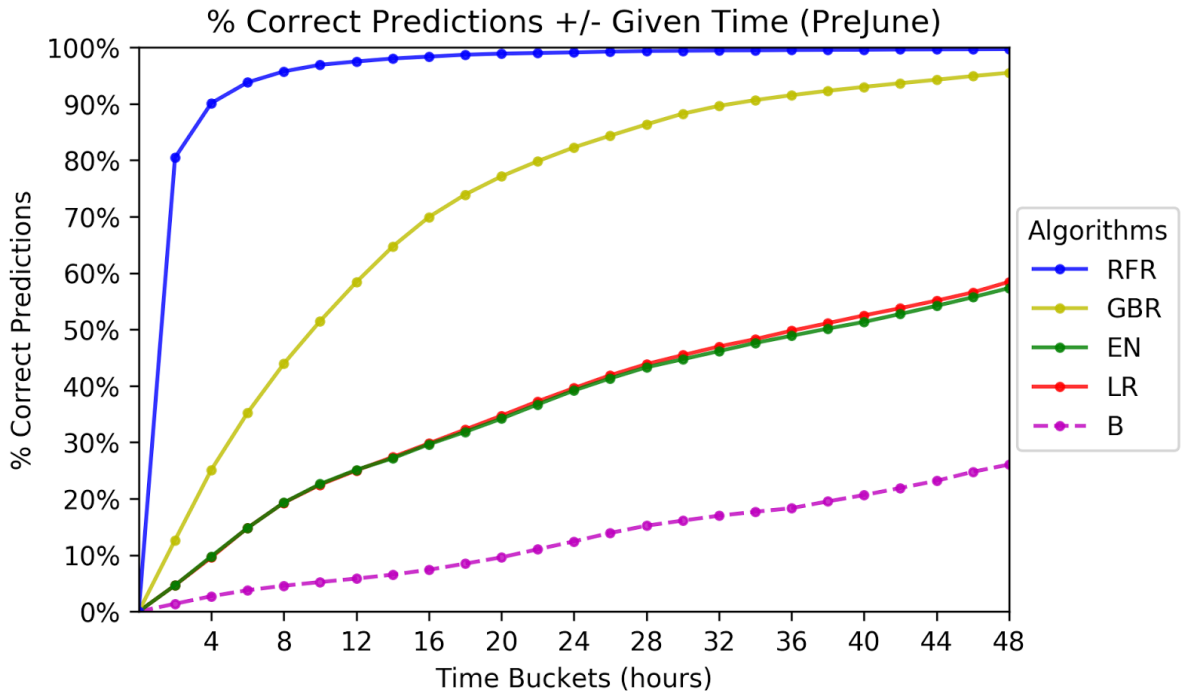

Figure 44: Correct predictions within given hours using pre-June training data

In the testing stage, however, RFR and GBR performed much worse than usual, but still better than LR and EN and much better than the Baseline, this can be seen in Figure 45. The shape of the RFR and GBR curves match much more closely those of LR and EN than compared to previous experiments, with each of them rising in a logarithmic fashion.

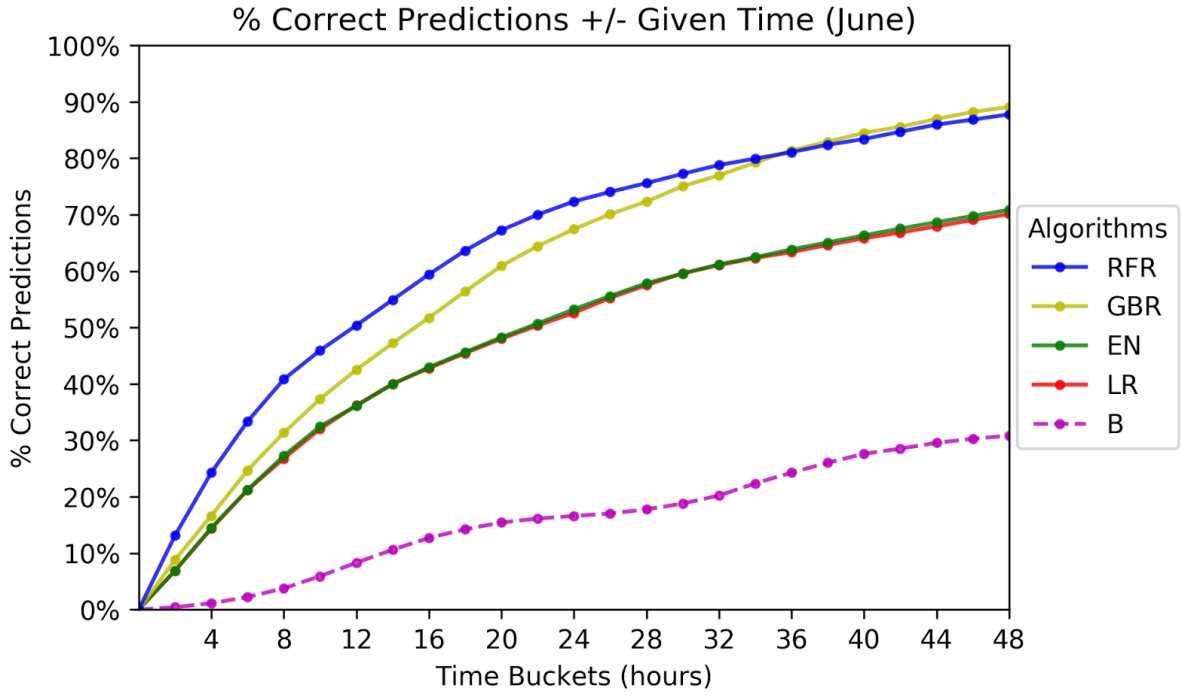

Figure 45: Correct predictions within given hours using June testing data

Plots for EN and RFR showing the predicted versus actual Time Taken for both training and testing stages can be seen in Appendix 6.

As seen in Figure 46, both LR and EN performed better when used to predict values for the unseen June data than for the training pre-June data. But GBR, and particularly RFR, performed much worse when predicting values for the unseen June data. This could be attributed to both tree-ensemble algorithms overfitting the training data and then not generalizing well to data from an unseen month which follows a different distribution.

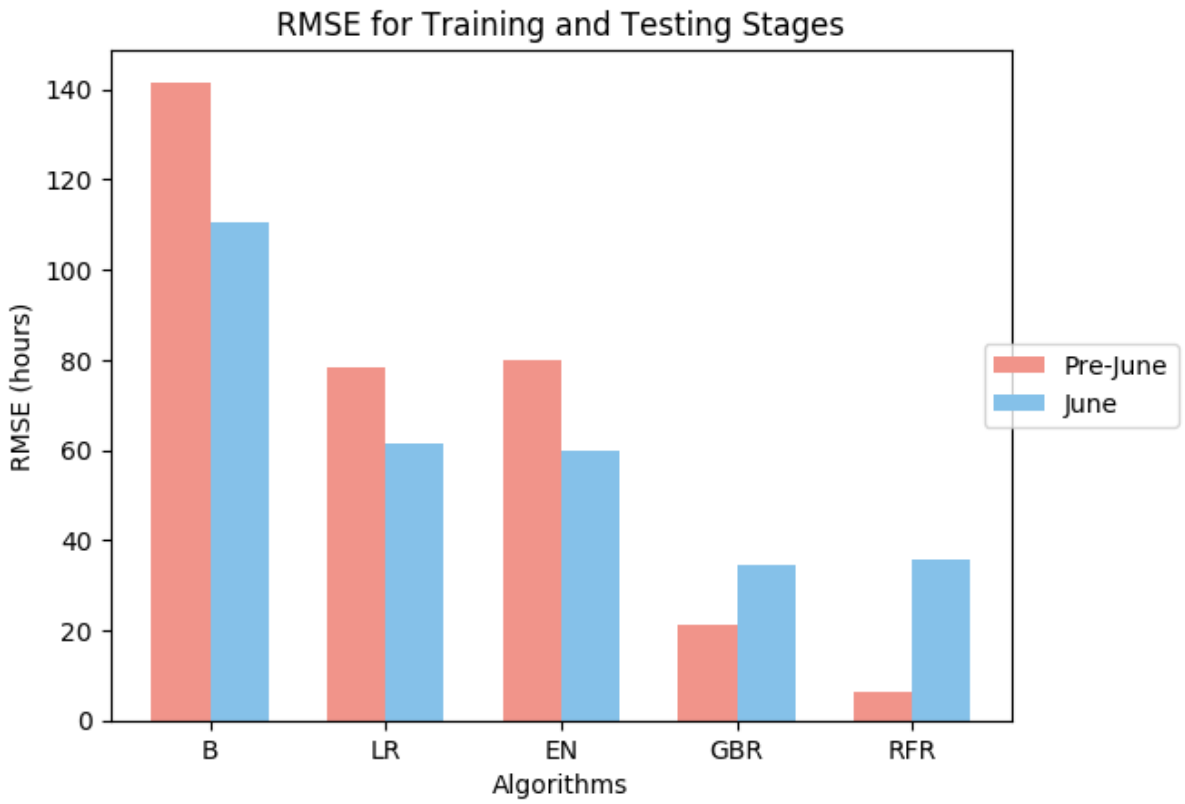

Figure 46: Pre-June training RMSE compared with June testing RMSE

As seen in Figure 47, when predicting the Time Taken for unseen June data, RFR has not performed to its usual high standards. The LR and EN testing results however have returned much better results than those seen in the main B4 experiment, with a RMSE around 10 hours smaller.

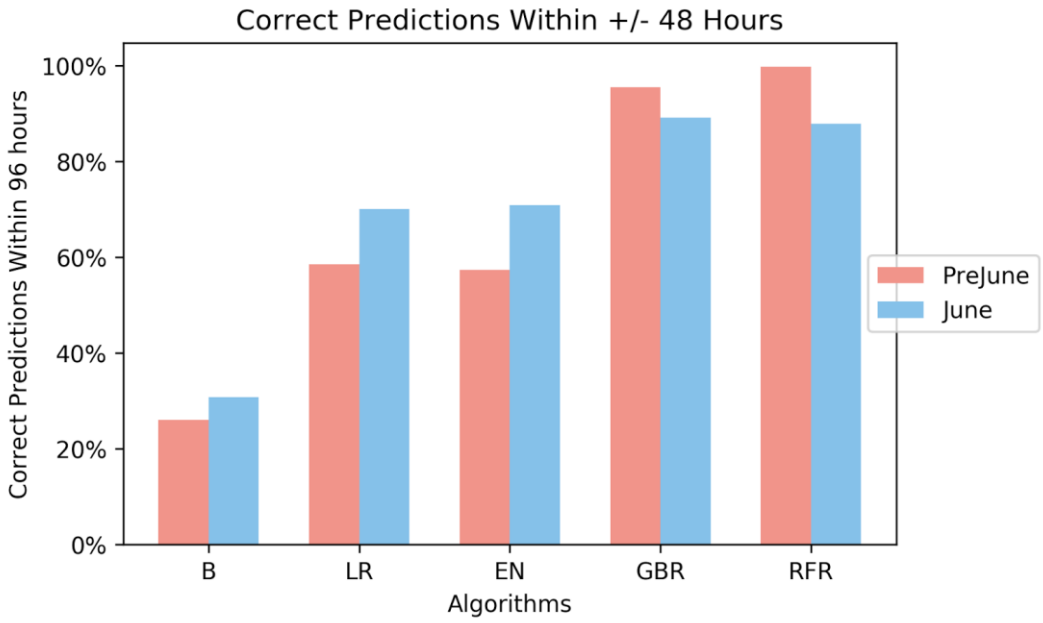

Figure 47: Correct predictions to +-48 hours for Pre-June and June data

9.2.2 Experiment C2: Testing with Unseen July Data

This experiment used a similar setup to experiment C1, however, pre-July data was used in the training stage and July data was used in the testing stage. Once again, each of the models performed excellently at predicting values for the training data, however, in the testing stage the results were quite poor.

As seen in Figure 48, all the machine learning algorithms were outperformed by the baseline when used to predict values in the unseen July data.

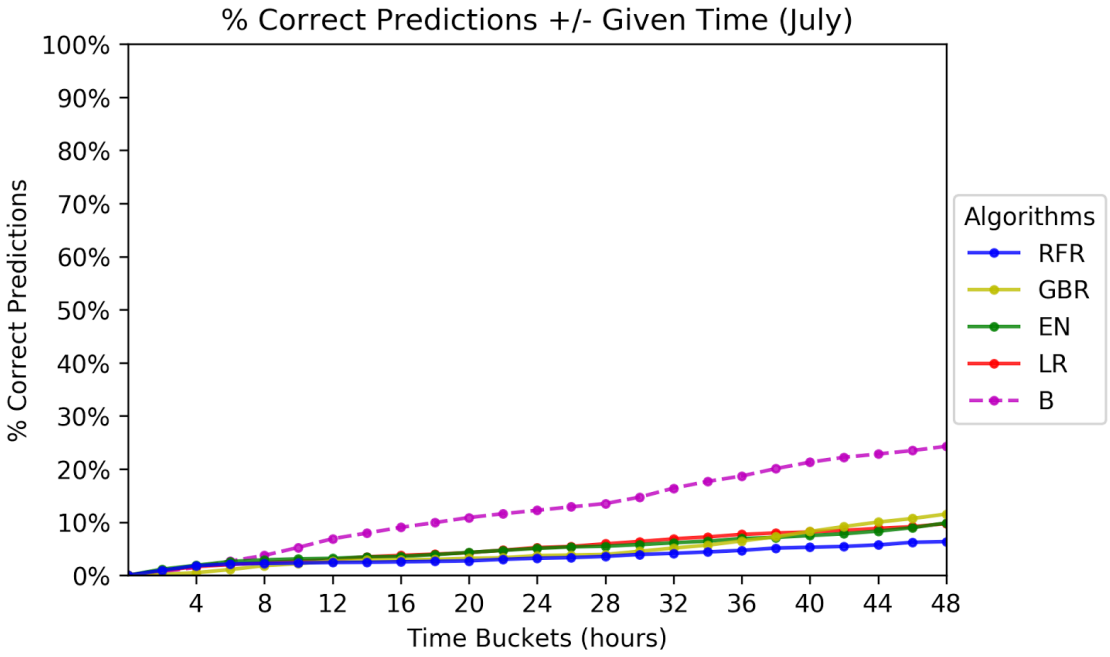

Figure 48: Correct predictions within given hours using July testing data

Plots for EN and RFR showing the predicted versus actual Time Taken for both training and testing stages can be seen in Appendix 7.

As seen in Figure 49, when a case is created in July it has quite a large RMSE relative to cases created in June and Figure 50 reaffirms this. At around 183 days into the year (i.e. the 1st of July), the RMSE becomes much larger than it was over the previous 80 days.

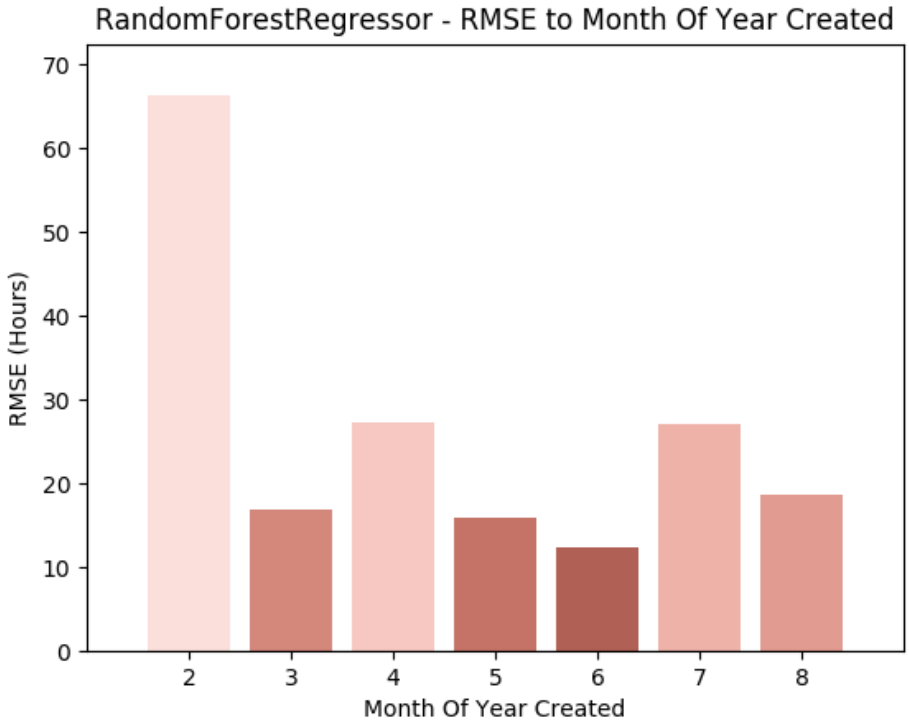

Figure 49: RMSE for each month of the year (main B4 experiment results)

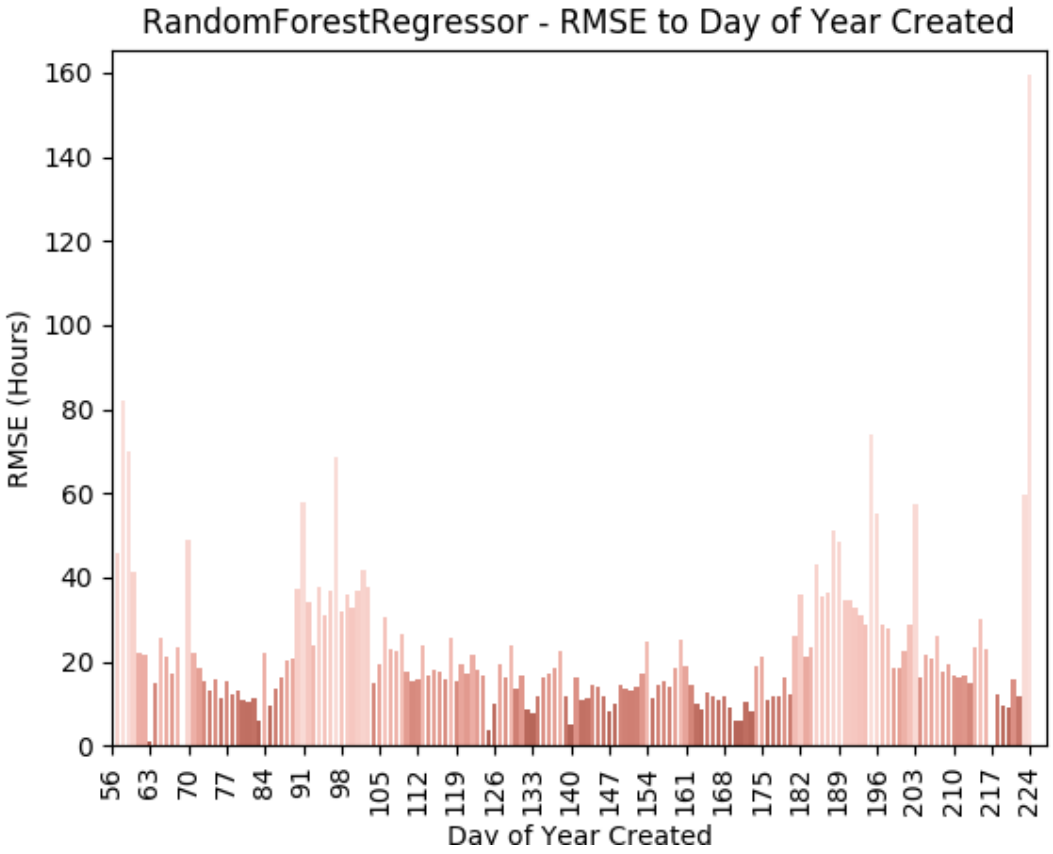

Figure 50: RMSE for each day of the year (main B4 experiment results)

As can be seen in Figure 51, none of the machine learning algorithms could accurately predict this unseen data. Each algorithm had a similar RMSE to the baseline of around 120 hours.

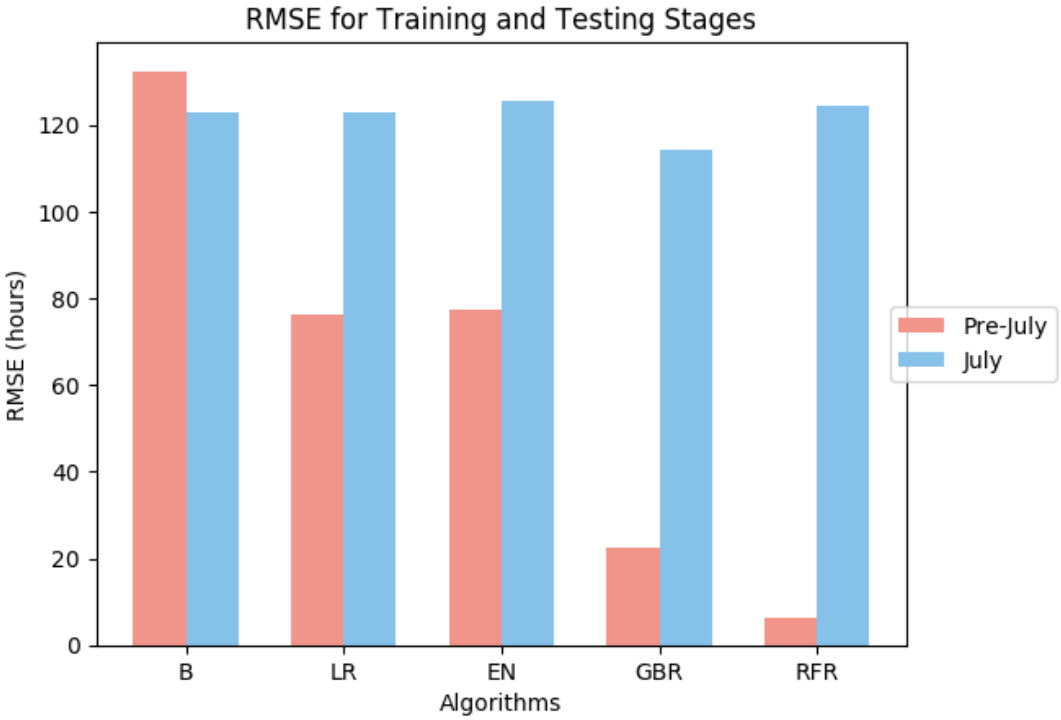

Figure 51: Pre-July training RMSE compared with July testing RMSE

It's clear from Figure 52 that, the machine learning algorithms which performed well in the training stage, are outperformed by the baseline in the testing stage. Thus, for this experiment, the simplest estimator generalises best to the unseen data.

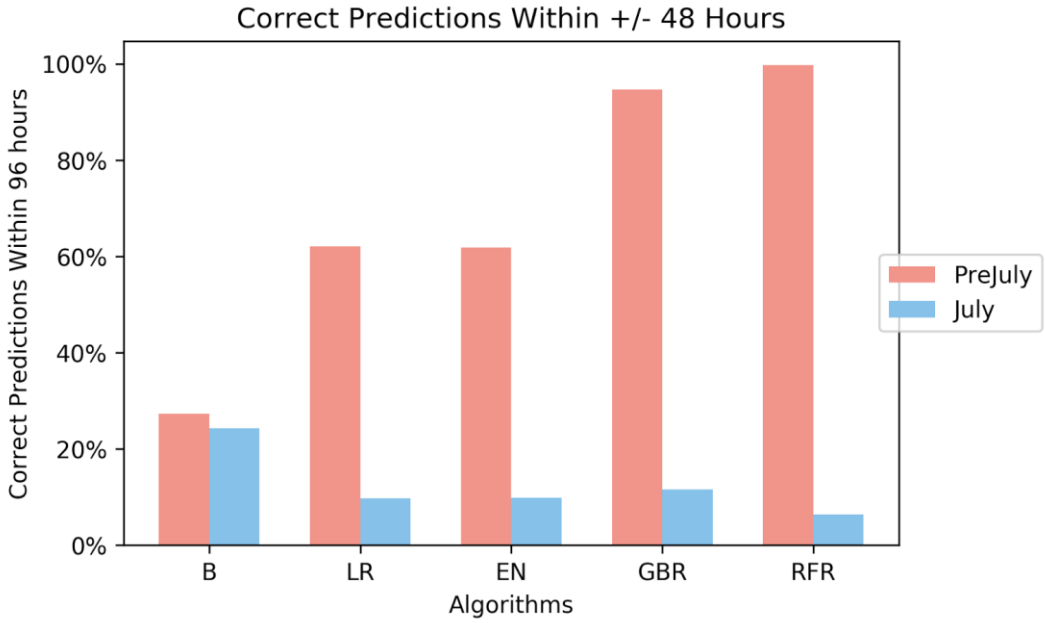

Figure 52: Predictions within 96 hours for Pre-July and July data

9.3 Modelling Considerations

9.3.1 Preprocessing Run Time

The preprocessing step including all the variable additions has a runtime of O(n^2) where n is the number of observations in the training dataset. This runtime is a result of operations requiring comparison of each data point with all other data points in the data set. This aspect was not a big concern during The Practicum as the number of observations represented roughly six months of data and preprocessing ran on modern computers within 10 minutes.

It is worth noting that many years of data would cause the runtime to become significant and one method of overcoming this would be to preprocess a random sample of all available data. The best modelling algorithm, Random Forest Regression, will also be sensitive to increased numbers of observations and random sampling should keep modelling run time reasonable. However, random sampling would reduce the overall predictive accuracy of the model so a trade-off must be made between accuracy and runtime.

9.3.2 Sample Size

Four datasets were made available during The Practicum. The first dataset represented two months of observations and the final dataset represented six months of data as more data became available. Table 7 shows the impact that an increased number of observations had on accuracy of the most predictive model (as described in experiment B4). While a minimum required sample size has not been calculated, more observations generally result in better predictions. This did not hold true in the test cases shown below. However, the latest dataset would most likely provide a more robust prediction for future data.

The seasonality aspect of the data most likely negatively affected performance as more seasonal data and therefore heteroscedasticity was included as the sample size increased. The seconds to end of year variable in combination with multiple years of data should overcome this shortfall. The larger RMSE errors could also be attributed to the RMSE metric in general getting larger as the sample size increases. However, as seen in Appendix 8, when the R Squared value is used instead of the RMSE, as the sample size increases, the R Squared score decreases meaning that the larger sample size is negatively affecting the performance of the model irrespective of whether RMSE is being used to evaluate each model's performance.

Table 7: Comparison of dataset error metrics

Test Results	**Dataset (Named by date pulled)**			
RMSE (std) in hours	20/04/2017	23/06/2017	25/07/2017	15/08/2017
No. Observations	4583	19924	25986	28204
EN	54.99 (3.19)	61.64 (2.22)	63.35 (1.10)	75.56 (2.12)
GBR	17.00 (1.00)	23.07 (0.79)	27.22 (0.37)	31.69 (0.89)
LR	55.70 (5.70)	60.18 (1.76)	62.21 (1.05)	74.24 (2.22)
RFR	11.09 (0.87)	11.42 (0.75)	15.02 (0.59)	19.21 (0.93)
Baseline	137.99	121.68	117.52	129.46

9.4 Further research

There exists an opportunity to research and design for Microsoft a new simulation system which could forecast when and how many cases will be created. This model, in conjunction with the recommended model from The Practicum, could be used to provide Microsoft's leadership with estimates of not only how many cases will be created and when, but also how long they will take to process. Together these models could be used to great effect in scenario analysis and could greatly assist Microsoft's management in allocating staff resources ahead of important deadlines such as the end of financial quarters.

Microsoft and many other companies are most interested in predicting processing times during periods before key dates. Thus, further research could consider the creation of a model which only looks at data taken from the last 4 business days of each month. A specialised model, that is only trained using cases created from the last 4 business days of each month, could be more predictive than a model trained using all data because it would not have to learn relationships and patterns in the data which occur outside of these periods.

Chapter 10 - Conclusion

The objective of The Practicum was to provide a recommendation to Microsoft on how to predict the time required to process ongoing work flow items. Chapter 2: Business Understanding and Chapter 3: Data Understanding provided a knowledge base that predictive models were built on. This understanding will assist Microsoft in following the recommendations outlined below.

10.1 Most Predictive Model

The preprocessing methodology in experiment B4 along with modelling using Random Forest Regression provided the strongest predictions. This method includes the preprocessing steps of variable filtering, transformation and variable generation.

Several experiments were carried out to test the model under plausible scenarios. Using the techniques outlined, the target figure of 90% correct predictions to within an error margin of +/- 48 hours on unseen data was exceeded. This model provides a significant improvement in comparison to Microsoft's previous method of prediction, using the mean case time. Figure 53 is a cumulative graph which summarises the Random Forest Regression prediction accuracy in comparison to the baseline within a given margin of error as shown on the horizontal axis.

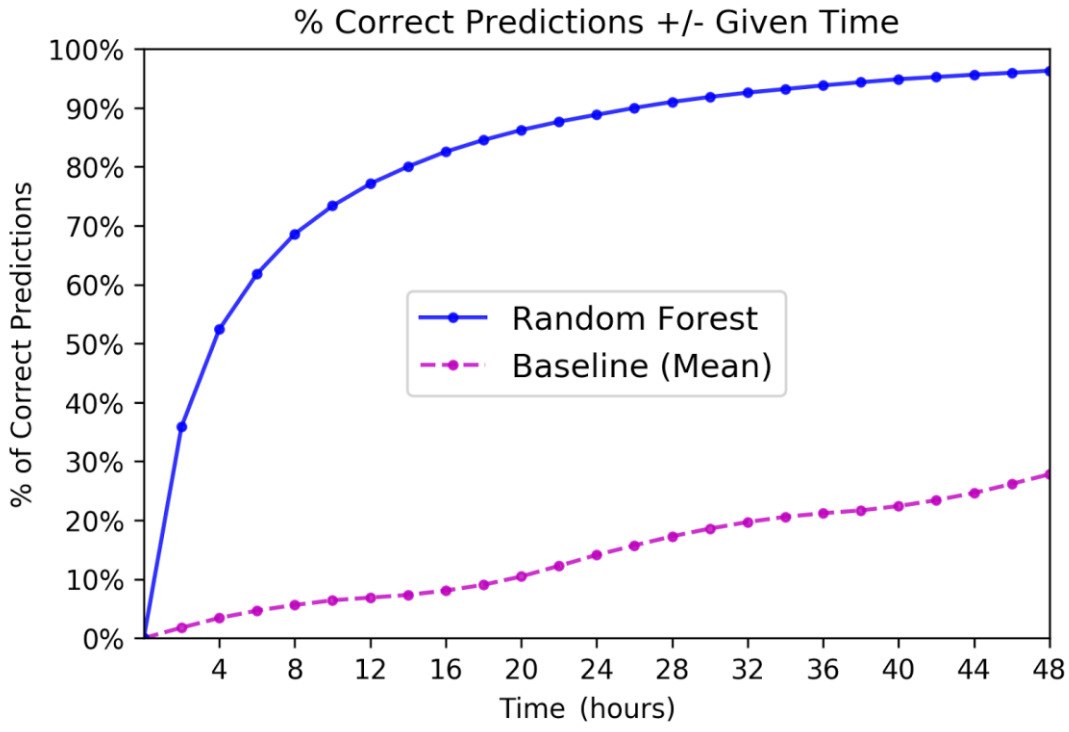

Figure 53: Most predictive model - correct predictions within given margins of error

Contrary to initial impressions, predictive power did not radically improve as a case progressed through processing and more data was available.

The number of open cases at the time a new case was entered onto the system is the strongest indicator. The top five predictive variables are listed below:

1. "Concurrent_open_cases"
2. "Rolling_Std"
3. "Seconds_left_Qtr"
4. "Seconds_left_Month"
5. "Rolling_Mean"

The rolling standard deviation and rolling mean are calculated for each case from the previous 10 created case's Time Taken values.

Unfortunately, this model cannot be used in practice. This is because it makes use of several variables which are not available while a case is being worked on which means making predictions for ongoing cases is not possible. However, many of the most predictive variables are available at the time a new case is created which allows for the creation of a recommended model which can make real time predictions for ongoing cases.

10.2 Recommended Model

As shown in Chapter 9: Results & Discussion, when only the 6 Minimum variables which can be constructed from each case's Created_On timestamp are used to train the RFR model, the results are almost as good as when all variables from the most predictive model are used. This means that a powerful model which can predict the completion times for ongoing cases is available to Microsoft. Once trained, this model can make predictions in real time, giving Microsoft's SDMs very accurate predictions for ongoing cases. This model provides a huge improvement over Microsoft's current estimator, mean case processing time.

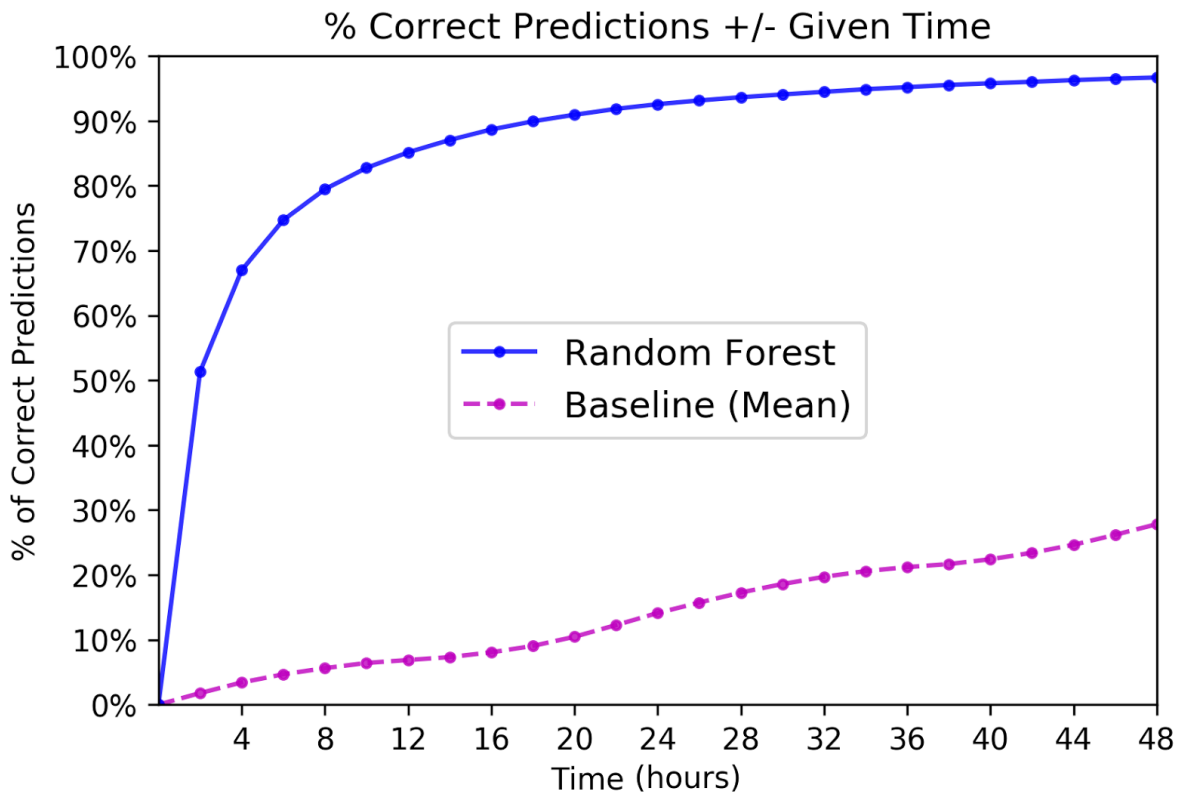

Figure 54: Recommended model - correct predictions within given margins of error

The recommended model can also be used to make Time Taken predictions in advance of cases being created. To do this, estimates of both the volume and timestamps of potential cases can be created using business domain knowledge or a new machine learning model to generate said cases could be formulated. These simulated cases can be fed into the recommended RFR model and the Time Taken predictions returned by it can be used to forecast how long they will take to be processed. This information can then be used by Microsoft's SDMs to manage staff resources in advance of deadlines so that case processing happens reliably before each deadline.

The simulated case study in experiment C2 which trained the model using pre-July data and tested the model using July data showed that the model needs to be trained by cases collected in a month for it to make accurate predictions for that month. Combining this knowledge with that gained from other experiments we can infer that when a full year of data has been generated and used to train the model, it will be able to accurately predict the Time Taken for cases created on any date. However, until such data has been generated, the RFR algorithm could be outperformed by a simpler LR algorithm or Baseline estimation.

10.3 CRISP-DM Iterations

CRISP-DM was used to structure the written The Practicum and programming iterations. Five iterations were completed with deeper business and data understandings, more advanced preprocessing and more powerful modelling used at each iteration. Microsoft should continue this process even after implementing the recommended model.

10.3.1 Seconds to End of Year

The seconds to end of year variable was a powerful predictor and it is recommended that this variable is included in the prediction system once at least one full year of data is available.

10.3.2 Staff Resources Variable

The recommended prediction model gives a snapshot prediction based on Time Taken for similar historic cases in the dataset. While this may be useful to predict if the cases in the system will be completed on time, it will not confirm if varying staff resources will impact the transaction times. In addition to this shortfall, prior staff levels have not been accounted for which means there may have been occasions where high or low staff levels influenced processing times and this is not modelled. Lastly, if resources are temporarily increased at the end of each quarter, the model will assume similar resource cycles which reduces the power of future predictions.

The staff resource data was unavailable for The Practicum. It is recommended that any future models include a number of generated variables representing staff numbers. Two examples are a variable representing the number of staff working on the day a case is created and the average number of staff working while a case was open. This improvement should overcome the shortfall as described and provide a more rounded model that Microsoft could implement.

This staff count variable could also be used for scenario planning to determine optimal staff numbers.

GitHub link: https://github.com/K-Ellis/Predicting-Transaction-Times

Appendices

Appendix 1: Data Understanding Supplement

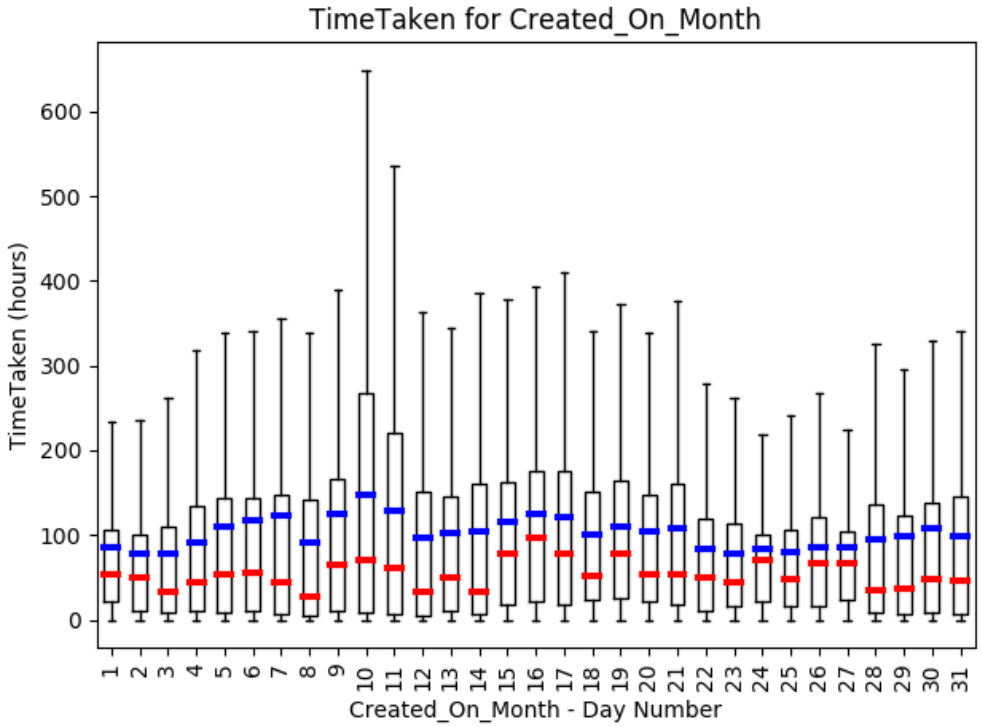

Figure 55: Created_On - day of the month boxplots

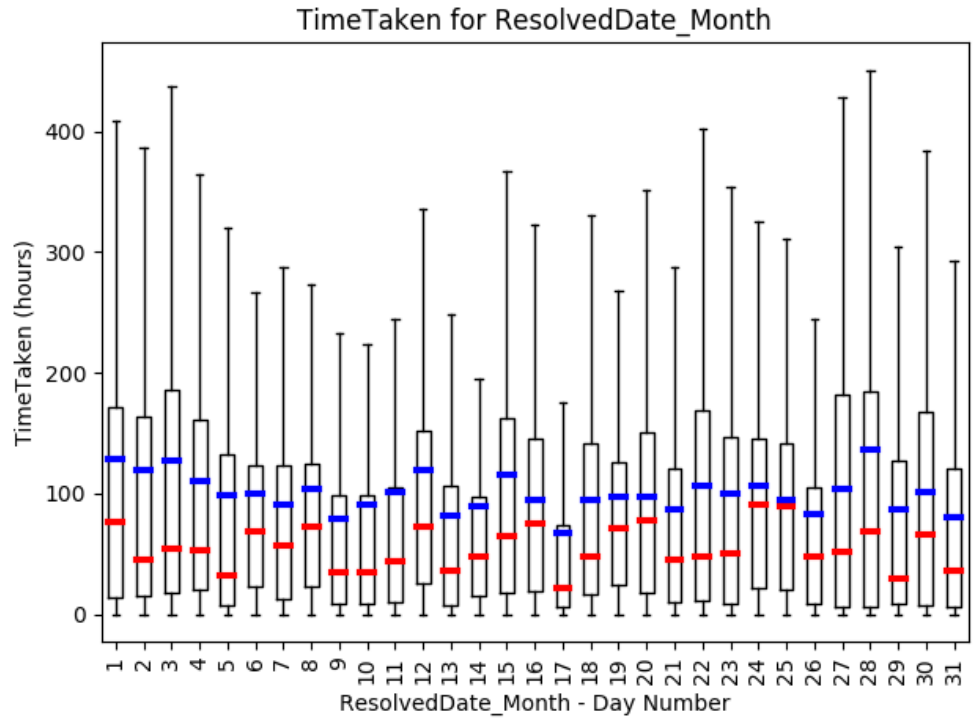

Figure 56: ResolvedDate - day of the month boxplots

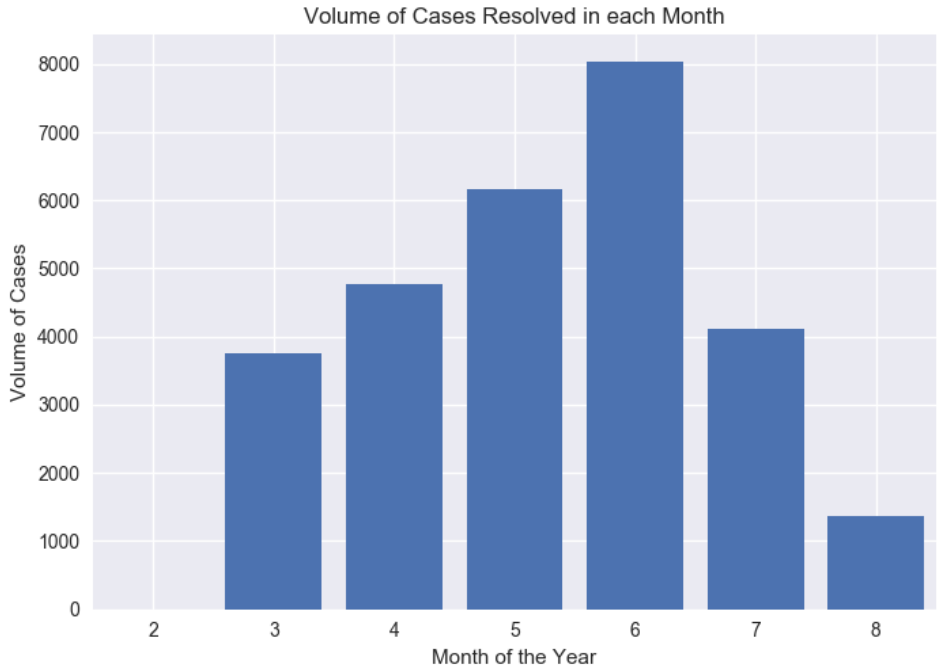

Figure 57: Volume of cases resolved each month

As seen in Figure 57, none of the 34 cases created in February were resolved in that month. The largest volume of cases was resolved in June.

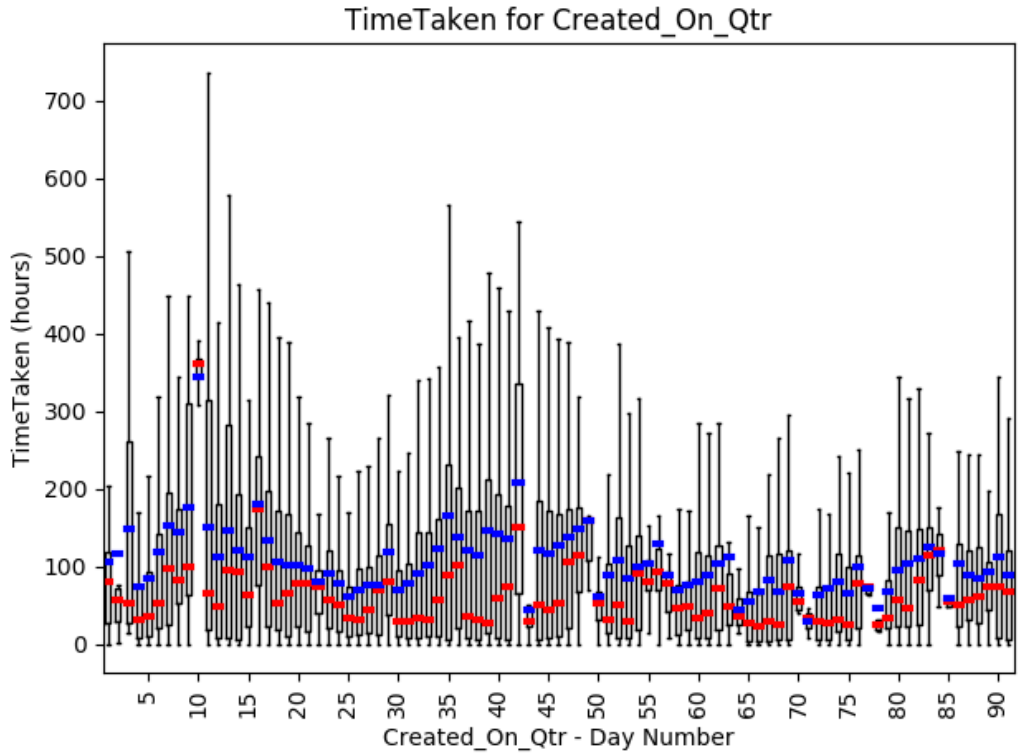

Figure 58: Created_On - day of the quarter boxplots

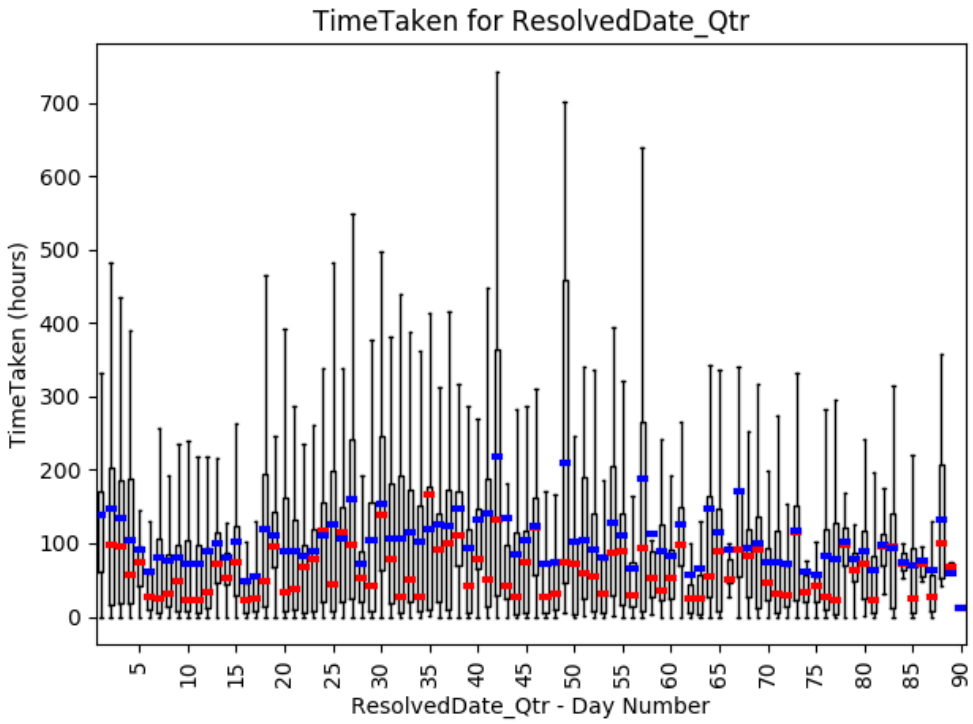

Figure 59: ResolvedDate - day of the quarter boxplots

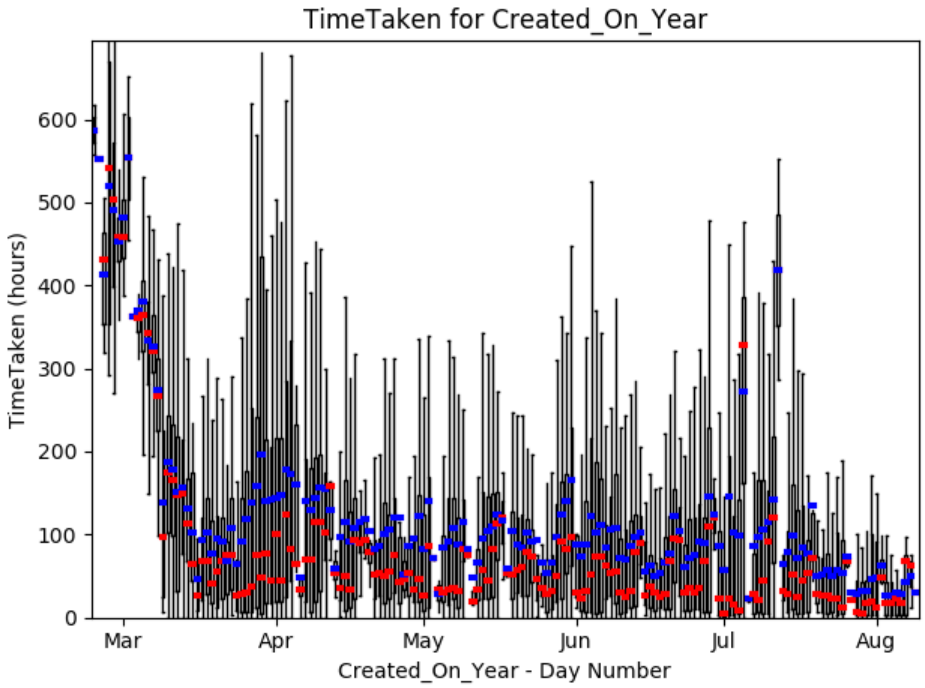

Figure 60: Created_On - day of the year boxplots

Figure 60 shows the day numbers from when the data from COSMIC was first made available (February), until August. As can be seen, the 34 cases created in February and cases created at

the beginning of March had a much larger average Time Taken than those created later in the year.

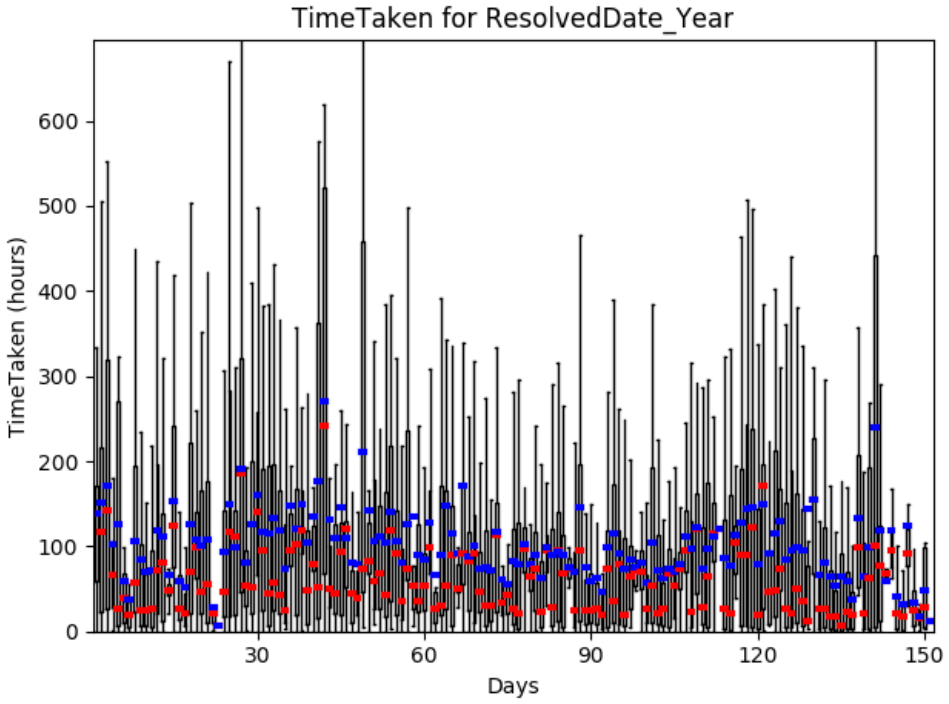

Figure 61: ResolvedDate - day of the year boxplots

Appendix 2: Generated Variables Average Time Taken Plots

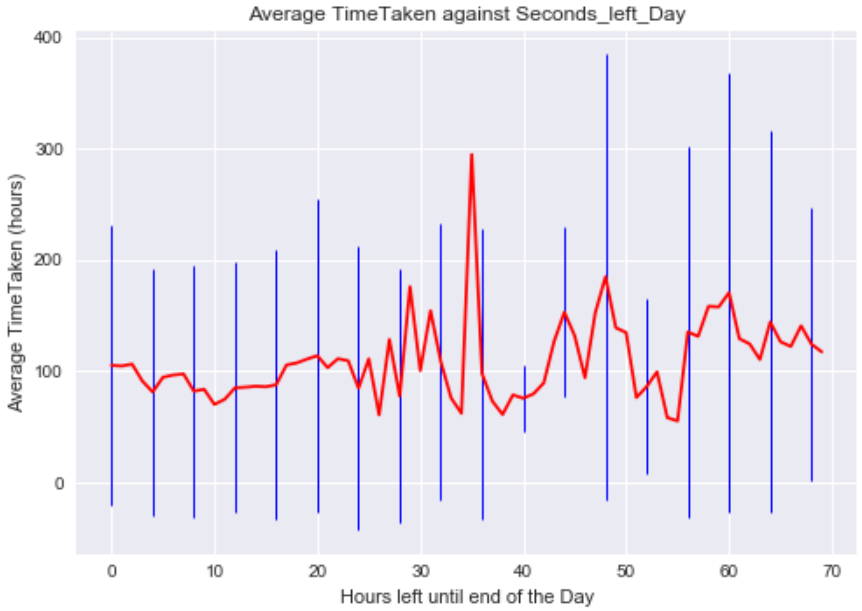

Figure 62: Average Time Taken against seconds left day

As can be seen in Figure 62, there is little correlation between the average Time Taken and the seconds until the end of the day. Shown in blue is the standard deviation the Time Taken and the error is very large throughout.

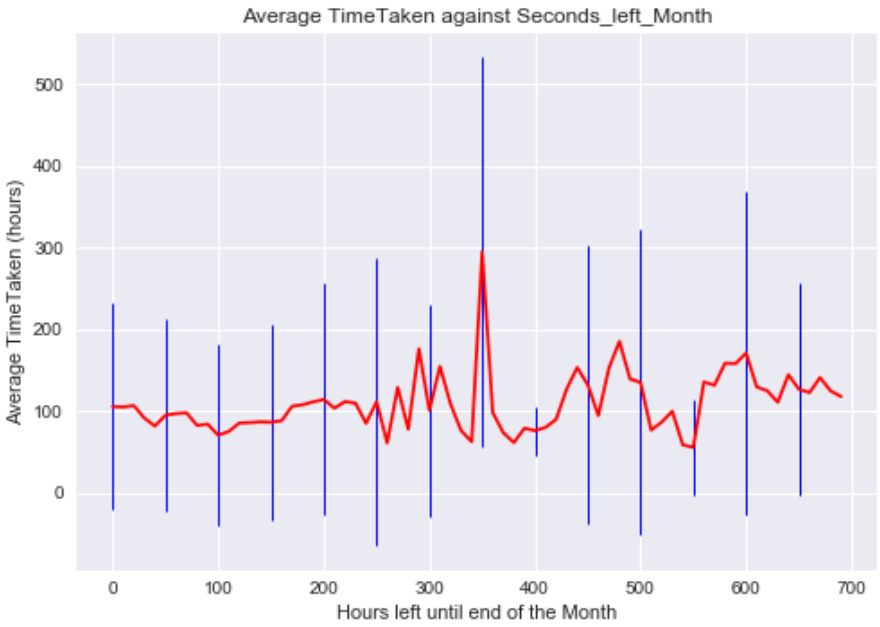

Figure 63: Average Time Taken against seconds left month

Seen in Figure 63, there is little correlation between the average Time Taken and the seconds until the end of the month. The standard deviation error bars are also quite large.

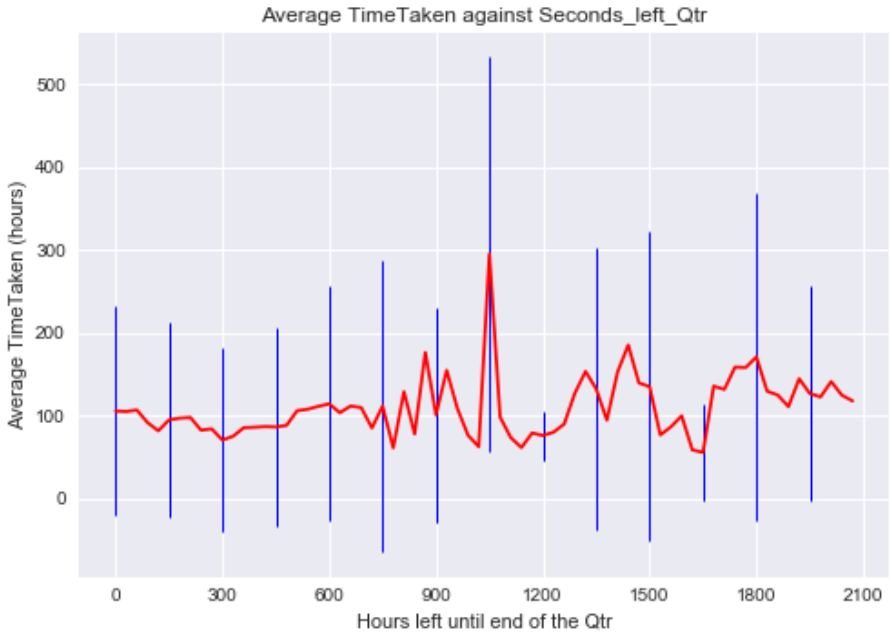

Figure 64: Average Time Taken against seconds left quarter

Again, as seen in Figure 64, there is little correlation between the average Time Taken and the seconds until the end of the quarter. The standard deviation error bars are also quite large.

Appendix 3: Variable Explanations Tables

The following tables describe the information that was available in the raw data files, the business understanding in the form of Microsoft's variable use, the preprocessing that was applied and the reasoning.

Vw_Incident Worksheet

Table 8: Vw_Incident worksheet variables

Variable Name	Variable Use	Preprocessing	Reasoning
TicketNumber	Unique Identification number for a case	Use to find hold and audit sinformation - then delete	Need for case ID
IncidentId	System identifier for the case	Delete column	Use TicketNumber for ID
Created_On	Created in COSMIC by BPO team	Use to calculate time taken – then delete	Needed for time taken only
Receiveddate	Operations Centre received the pack	Delete column	Not needed since we can calculate time
ResolvedDate	Date the case was resolved	Use to calculate time taken – then delete	Needed for time taken only
Queue	The location of case at time of pulling data. Used to assign to speciality teams to be worked on	Combine types, One hot encoding	Too many options for direct one hot

StatusReason	Users change the status. Outcome of case at time of pulling data. Online includes Cases which have been resolved at the time of pulling the data.	Filter out "Rejected", One hot	Not interested in rejected
Priority	Case priority	Mapping	Ordinal variable
ValidCase	Binary - if a case was valid	Filter by 1, Delete column (min_variable_types function will clear this)	Only interested in 1 entries
BusinessFunction	-	Delete column (min_variable_types, function will clear this)	Only 1 variable type
LineOfBusiness	-	Delete column (min_variable_types, function will clear this)	Only 1 variable type
Program	-	Filter by "Enterprise", Delete column (min_variable_types function will clear this)	Only want "Enterprise" entries
CaseType	-	Delete column (min_variable_types, function will clear this)	Only 1 variable type
CaseSubTypes	-	Delete column (min_variable_typ	Only 1 variable type

		es, function will clear this)	
Reason	-	Delete column (min_variable_types, function will clear this)	Only 1 variable type
SubReason	Lower level detail of Reason	One hot	
SubSubReason	-	Delete column (min_entries function will clear this)	No entries present
CountrySource	Where was the case submitted from	Combine types One hot encoding	Important but need to place in buckets
CountryProcessed	Where is case being processed	Combine types, One hot encoding	Important but need to place in buckets
SalesLocation	Where is sale being made to	Combine types, One hot encoding	Important but need to place in buckets
ROCName	Regional operating centre name	One hot encoding	
CaseRevenue	In local currency	Delete column	
CurrencyName	-	Delete column	We have all in USD, can delete others
IsoCurrencyCode	-	Delete column	We have all in USD, can delete others
RevenueImpactAmount	-	Delete column	Mostly NULL entries
IsGovernment	-	Keep as is	

IsAudited	-	Delete column	Mostly 0 entries
Language	-	Delete column (min_variable_types function will clear this)	Only 1 variable type (after filtering)
LanguageName	-	Filter by "English", Delete column (min_variable_types function will clear this)	Only want "English" entries
sourcesystem	Where the case came from - free text field for user. Which Microsoft IT system was used	One hot encoding	
caseOriginCode	-	Delete column	Do not know what this does
Source	Where the case came from - drop down list.	One hot encoding	
pendingemails	-	Delete column	Mostly 0 entries
RelatedCases	-	Delete column	Don't know how else to include this
Auditresult	-	Delete column	Mostly 0 entries
PendingRevenue	-	Delete column	Mostly 0 entries
Requestspercase	-	Delete column	Mostly 1 entries
Totalbillabletime	-	Delete column	Mostly 0 entries
Totaltime	-	Delete column	Mostly 0 entries
CreditAmount	-	Delete column	Mostly NULL entries
DebitAmount	-	Delete column	Mostly NULL entries
OrderAmount	-	Delete column	Not enough entries

InvoiceAmount	-	Delete column	Mostly NULL entries
Deleted	-	Delete column	No entries
TotalIdleTime	Internally calculated time - not important	Delete column	Internal measurement
TotalWaitTime	Internally calculated time - not important	Delete column	Internal measurement
WorkbenchGroup	Turned off now - disregard	Delete	Told to ignore
Workbench	Turned off now - disregard	Delete	Told to ignore
RejectionReason	-	Delete column	Mostly NULL entries
RejectionSubReason	-	Delete column	Mostly NULL entries
PackageNumber	-	Delete column	Mostly NULL entries
RequiredThreshold	-	Delete column	Mostly 0 entries
Slipped	-	Delete column	Mostly 0 entries
DefectiveCase	-	Delete column	Mostly 0 entries
OLSRevenue	Indicates if Online Services Revenue associated with case	Keep as is	MS place low emphasis
Revenutype	Type of revenue that MS will receive. Eg immediate or future value	One hot encoding	
ProcessName	-	Delete column	Only 1 variable type
StageName	Stages sox cases pass through. If not sox, this stays as "OpsIn"	Mapping	Ordinal variable (stages in email)

AmountinUSD	-	Keep as is, but normalise	Important variable
IsMagnumCase	MS Sox tool to be used or not	Keep as is	
IsSignature	If a customer signature is required	Keep as is	
Complexity	-	Mapping	Ordinal variable
Numberofreactivations	-	Keep as is	Not enough entries
NumberofChildIncidents	-	Delete column	Mostly NULL entries
ParentCase	-	Delete column	Mostly NULL entries
Referencesystem	-	Delete column	Not enough entries
StateCode	-	Delete column	Only 1 variable type
Isrevenueimpacting	-	Delete column	Mostly 0 entries
IsSOXCase	Binary: A legal document requirement	Delete rows with no entry - this is important, will always be available going forward	Important variable. Can't assume NULL means 0

Vw_HoldActivity

Table 9: Vw_HoldActivity variables

Variable Name	Variable Use	Preprocessing	Reasoning
TicketNumber	Links to case ID from Incident datasheet	Use for linking - then delete	Need for case ID
ActivityId	System ID which links to vw_Incident Incidentid column	Delete column	Use TicketNumber for ID

RegardingObjectId	-	Delete column	Use TicketNumber for ID
HoldDuration	-	Keep as is	Use for time
Statuscode	-	Delete column	Not used at CRISP-DM Iteration 5
Statecode	-	Delete column	Duplicate of Statuscode
TimeZoneRuleVersionNumber	-	Delete column	Not used at CRISP-DM Iteration 5
HoldTypeName	-	One hot encoding	
Reason	-	Delete column	Not used at CRISP-DM Iteration 5
AssignedToGroup	-	One hot encoding	
StartTime	-	Delete column	Use HoldDuration for time
EndTime	-	Delete column	Use HoldDuration for time
IsThreeStrikeRule	-	Delete column	Only 1 variable type
HoldSubReason	-	Delete column	Mostly nulls
Modified_On	-	Delete column	Use HoldDuration for time
Deleted	-	Delete column	No entries

Vw_AuditHistory

Table 10: Vw_AuditHistory variables

Variable Name	**Variable Use**	**Preprocessing**	**Reasoning**
TicketNumber	Links to case ID	Use for linking - then delete	Use TicketNumber for ID

AuditHistory Id	System ID	Delete column	Use TicketNumber for ID
AuditId	-	Delete column	Use TicketNumber for ID
EntityId	Also links to Incident	Delete column	Use TicketNumber for ID
EntityLogical Name	-	Delete column	Only 1 variable type
CaseId	-	Delete column	Use TicketNumber for ID
NewValue	New Value for Stage - vw_StageTable shows what the stageid stands for	Delete column	Not used at CRISP-DM Iteration 5
OldValue	stage updated from old value	Delete column	Not used at CRISP-DM Iteration 5
Action	whether case updated or created	Delete column	Not used at CRISP-DM Iteration 5
Attribute	-	Delete column (min_variable_types function will clear this)	Only 1 variable type
TimeStamp	-	Delete column	Use Created_on
Deleted	-	Delete column	No entries
TenantId	-	Delete column	Only 1 variable type
VersionNum ber	-	Delete column	Only 1 variable type

StateCode	-	Delete column	Only 1 variable type
Status	-	Delete column	Only 1 variable type
StatusCode	-	Delete column	Only 1 variable type
StatusReason	-	Delete column	Only 1 variable type
UTCConversionTimeZoneCode	-	Delete column	Not used at CRISP-DM Iteration 5
Created_On	-	Use for time, Delete column	
Modified_On	-	Delete column	Not used at CRISP-DM Iteration 5
ImportSequenceNumber	-	Delete column	No entries
OverriddenCreatedOn	-	Delete column	No entries

Appendix 4: Results Supplement

Mandatory Feature Importances

LR

"Variable Name" (Standardised Regression Coefficient) [Percentage of Importance]

1. "Queue_Broken" (9345.169404) [0.498007]
2. "Queue_E&E" (5042.139385) [0.268697]
3. "CountrySource_other" (758.078511) [0.040398]
4. "CountryProcessed_other" (471.140809) [0.025107]
5. "CountrySource_southamerica" (402.050313) [0.021425]
6. "CountryProcessed_southamerica" (271.154053) [0.014450]
7. "SalesLocation_southamerica" (-236.654743) [0.012611]
8. "Queue_LOC" (-234.653273) [0.012505]

9. "CountrySource_australia" (232.695186) [0.012400]
10. "Queue_APOC" (226.049167) [0.012046]
11. "CountrySource_northamerica" (224.589264) [0.011968]
12. "CountryProcessed_australia" (-176.990038) [0.009432]
13. "ROCName_APOC" (-167.244123) [0.008912]
14. "SalesLocation_other" (-158.842988) [0.008465]
15. "SalesLocation_northamerica" (-152.798067) [0.008143]
16. "CountrySource_europe" (135.679238) [0.007230]
17. "CountryProcessed_europe" (-109.725372) [0.005847]
18. "SalesLocation_australia" (-95.498910) [0.005089]
19. "CountrySource_asia" (72.227940) [0.003849]
20. "Queue_NAOC" (-51.145241) [0.002726]
21. "ROCName_EOC" (36.576659) [0.001949]
22. "Created_on_Weekend" (-35.753090) [0.001905]
23. "SalesLocation_europe" (-27.812971) [0.001482]
24. "Queue_EOC" (25.999948) [0.001386]
25. "SalesLocation_asia" (-24.555544) [0.001309]
26. "CountryProcessed_northamerica" (24.188682) [0.001289]
27. "CountryProcessed_asia" (-18.846855) [0.001004]
28. "Cases_created_within_past_8_hours" (-1.668074) [0.000089]
29. "Concurrent_open_cases" (-1.653758) [0.000088]
30. "Seconds_left_Month" (-1.373364) [0.000073]
31. "Seconds_left_Qtr" (-1.320948) [0.000070]
32. "Seconds_left_Day" (-0.879329) [0.000047]

EN

"Variable Name" (Standardised Regression Coefficient) [Percentage of Importance]

1. "Cases_created_within_past_8_hours" (-1.668506) [0.093727]
2. "Concurrent_open_cases" (-1.653783) [0.092900]
3. "Seconds_left_Month" (-1.373364) [0.077148]
4. "Seconds_left_Qtr" (-1.320948) [0.074203]
5. "CountryProcessed_northamerica" (-0.940664) [0.052841]
6. "SalesLocation_northamerica" (-0.887342) [0.049846]
7. "CountrySource_northamerica" (-0.886257) [0.049785]

8. "Seconds_left_Day" (-0.879329) [0.049396]
9. "Queue_NAOC" (-0.778403) [0.043726]
10. "ROCName_EOC" (-0.702767) [0.039477]
11. "Queue_EOC" (-0.702205) [0.039446]
12. "CountryProcessed_europe" (-0.611347) [0.034342]
13. "SalesLocation_europe" (-0.601558) [0.033792]
14. "CountrySource_europe" (-0.600519) [0.033734]
15. "CountryProcessed_asia" (-0.520555) [0.029242]
16. "SalesLocation_asia" (-0.468829) [0.026336]
17. "CountrySource_asia" (-0.466689) [0.026216]
18. "ROCName_APOC" (-0.465844) [0.026168]
19. "Queue_APOC" (-0.465731) [0.026162]
20. "Created_on_Weekend" (-0.460932) [0.025893]
21. "CountrySource_southamerica" (-0.231960) [0.013030]
22. "SalesLocation_southamerica" (-0.230514) [0.012949]
23. "CountrySource_australia" (-0.207203) [0.011640]
24. "SalesLocation_australia" (-0.207017) [0.011629]
25. "CountryProcessed_southamerica" (-0.148310) [0.008331]
26. "Queue_LOC" (-0.138525) [0.007782]
27. "CountryProcessed_australia" (-0.056264) [0.003161]
28. "CountrySource_other" (-0.034742) [0.001952]
29. "Queue_Broken" (-0.031524) [0.001771]
30. "CountryProcessed_other" (-0.027940) [0.001570]
31. "Queue_E&E" (-0.018833) [0.001058]
32. "SalesLocation_other" (-0.013316) [0.000748]

RFR

The importances for each variable used by Random Forest Regression were as follows:

1. "Concurrent_open_cases" (0.781232)
2. "Seconds_left_Qtr" (0.107140)
3. "Seconds_left_Month" (0.052490)
4. "Cases_created_within_past_8_hours" (0.033224)
5. "Seconds_left_Day" (0.010643)
6. "Created_on_Weekend" (0.003258)

7. "CountryProcessed_asia" (0.001527)
8. "Queue_NAOC" (0.001511)
9. "ROCName_APOC" (0.001148)
10. "CountryProcessed_northamerica" (0.001070)
11. "Queue_APOC" (0.000921)
12. "CountryProcessed_southamerica" (0.000756)
13. "Queue_EOC" (0.000633)
14. "CountrySource_asia" (0.000626)
15. "ROCName_EOC" (0.000537)
16. "CountrySource_northamerica" (0.000493)
17. "SalesLocation_asia" (0.000481)
18. "CountryProcessed_europe" (0.000409)
19. "SalesLocation_northamerica" (0.000386)
20. "SalesLocation_europe" (0.000296)
21. "CountrySource_europe" (0.000292)
22. "SalesLocation_australia" (0.000265)
23. "CountrySource_australia" (0.000261)
24. "CountrySource_southamerica" (0.000118)
25. "SalesLocation_southamerica" (0.000106)
26. "Queue_LOC" (0.000070)
27. "Queue_Broken" (0.000067)
28. "CountryProcessed_australia" (0.000018)
29. "CountrySource_other" (0.000009)
30. "Queue_E&E" (0.000008)
31. "CountryProcessed_other" (0.000005)
32. "SalesLocation_other" (0.000002)

GBR

The importances for each variable used by Gradient Boosting Regression were as follows:

1. "Concurrent_open_cases" (0.381622)
2. "Seconds_left_Qtr" (0.304125)
3. "Seconds_left_Month" (0.175916)
4. "Cases_created_within_past_8_hours" (0.093899)
5. "Seconds_left_Day" (0.016979)

6. "CountryProcessed_northamerica" (0.011929)
7. "CountryProcessed_asia" (0.004694)
8. "Created_on_Weekend" (0.003275)
9. "CountrySource_asia" (0.003049)
10. "CountryProcessed_southamerica" (0.002474)
11. "Queue_NAOC" (0.002039)
12. "SalesLocation_asia" (0.000000)
13. "CountryProcessed_europe" (0.000000)
14. "SalesLocation_southamerica" (0.000000)
15. "SalesLocation_other" (0.000000)
16. "SalesLocation_northamerica" (0.000000)
17. "SalesLocation_europe" (0.000000)
18. "SalesLocation_australia" (0.000000)
19. "ROCName_EOC" (0.000000)
20. "ROCName_APOC" (0.000000)
21. "Queue_EOC" (0.000000)
22. "Queue_LOC" (0.000000)
23. "Queue_E&E" (0.000000)
24. "CountryProcessed_australia" (0.000000)
25. "Queue_APOC" (0.000000)
26. "CountrySource_southamerica" (0.000000)
27. "CountrySource_other" (0.000000)
28. "CountrySource_northamerica" (0.000000)
29. "CountrySource_europe" (0.000000)
30. "CountrySource_australia" (0.000000)
31. "CountryProcessed_other" (0.000000)
32. "Queue_Broken" (0.000000)

Minimum Feature Importances

LR

"Variable Name" (Standardised Regression Coefficient) [Percentage of Importance]

1. "Created_on_Weekend" (-35.791730) [0.838464]
2. "Cases_created_within_past_8_hours" (-1.668117) [0.039078]
3. "Concurrent_open_cases" (-1.653791) [0.038742]

4. "Seconds_left_Month" (-1.373364) [0.032173]
5. "Seconds_left_Qtr" (-1.320948) [0.030945]
6. "Seconds_left_Day" (-0.879329) [0.020599]

EN

"Variable Name" (Standardised Regression Coefficient) [Percentage of Importance]

1. "Cases_created_within_past_8_hours" (-1.668506) [0.226796]
2. "Concurrent_open_cases" (-1.653783) [0.224795]
3. "Seconds_left_Month" (-1.373364) [0.186678]
4. "Seconds_left_Qtr" (-1.320948) [0.179553]
5. "Seconds_left_Day" (-0.879329) [0.119525]
6. "Created_on_Weekend" (-0.460932) [0.062653]

RFR

The importances for each variable used by Random Forest Regression were as follows:

1. "Concurrent_open_cases" (0.781975)
2. "Seconds_left_Qtr" (0.110456)
3. "Seconds_left_Month" (0.054460)
4. "Cases_created_within_past_8_hours" (0.034398)
5. "Seconds_left_Day" (0.015164)
6. "Created_on_Weekend" (0.003548)

GBR

The importances for each variable used by Gradient Boosting Regression were as follows:

1. "Concurrent_open_cases" (0.381623)
2. "Seconds_left_Qtr" (0.310758)
3. "Seconds_left_Month" (0.188527)
4. "Cases_created_within_past_8_hours" (0.078737)
5. "Seconds_left_Day" (0.032580)
6. "Created_on_Weekend" (0.007774)

Appendix 5: Minimum variables

Predicted versus Actual Time Taken

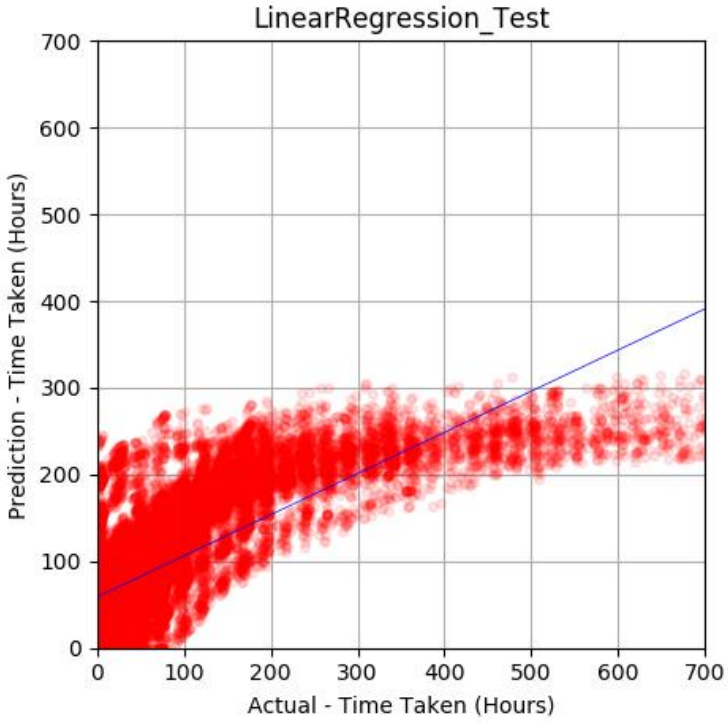

Figure 65: LR - Predicted versus Actual Time Taken

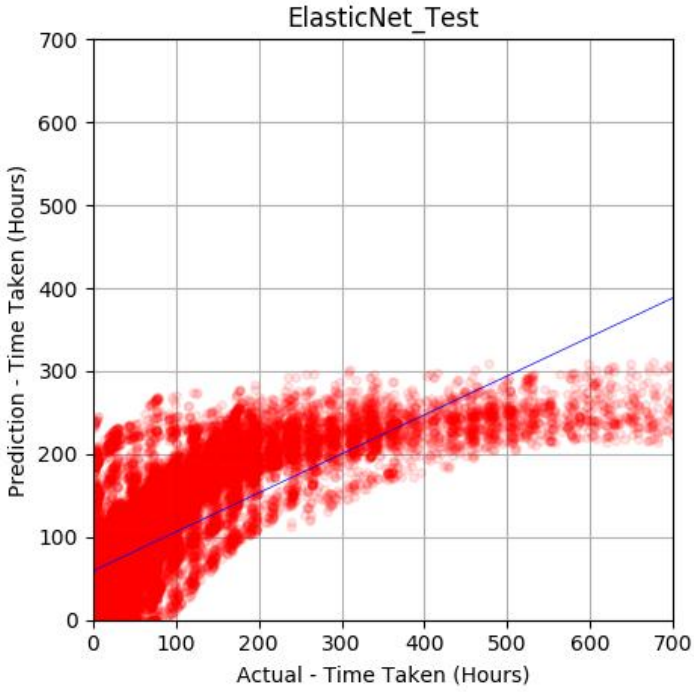

Figure 66: EN - Predicted versus Actual Time Taken

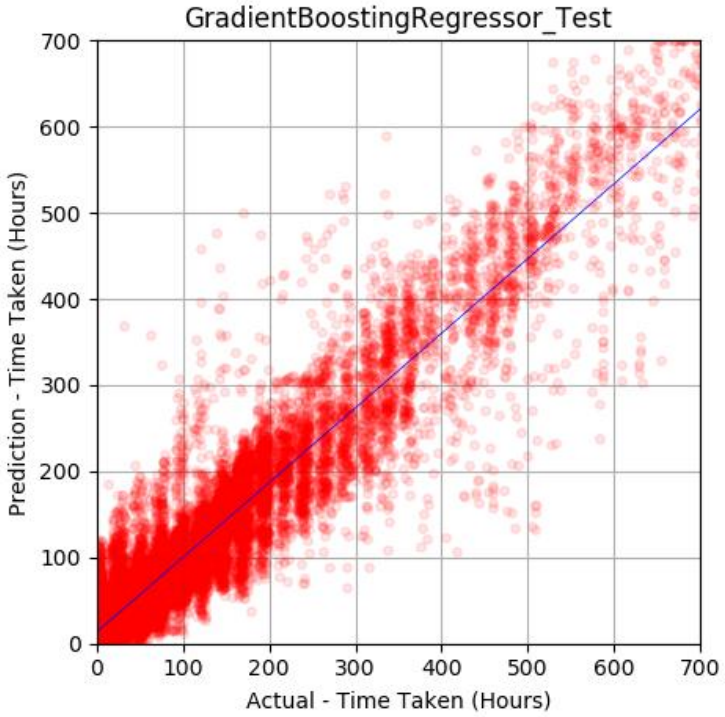

Figure 67: GBR - Predicted versus Actual Time Taken

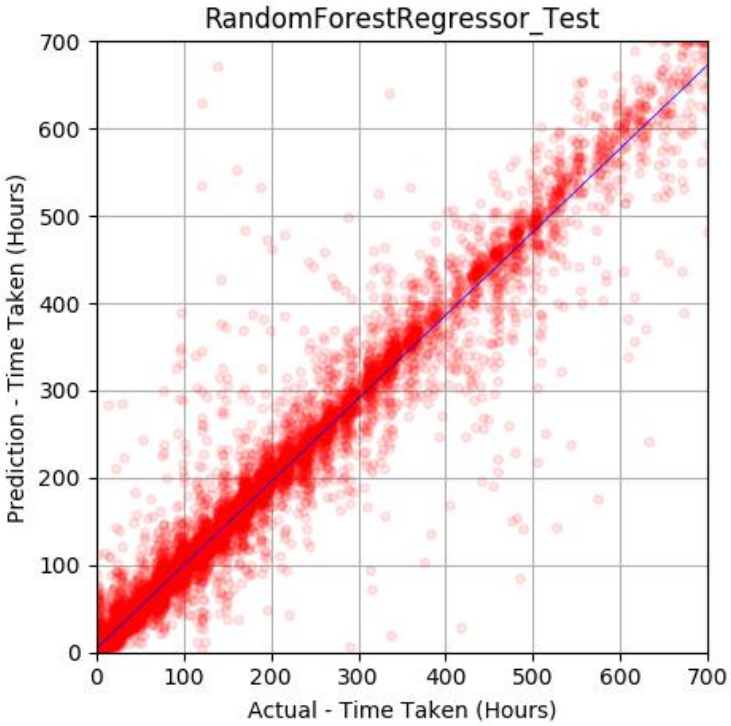

Figure 68: RFR - Predicted versus Actual Time Taken

Standardised Residual versus Predicted Time Taken

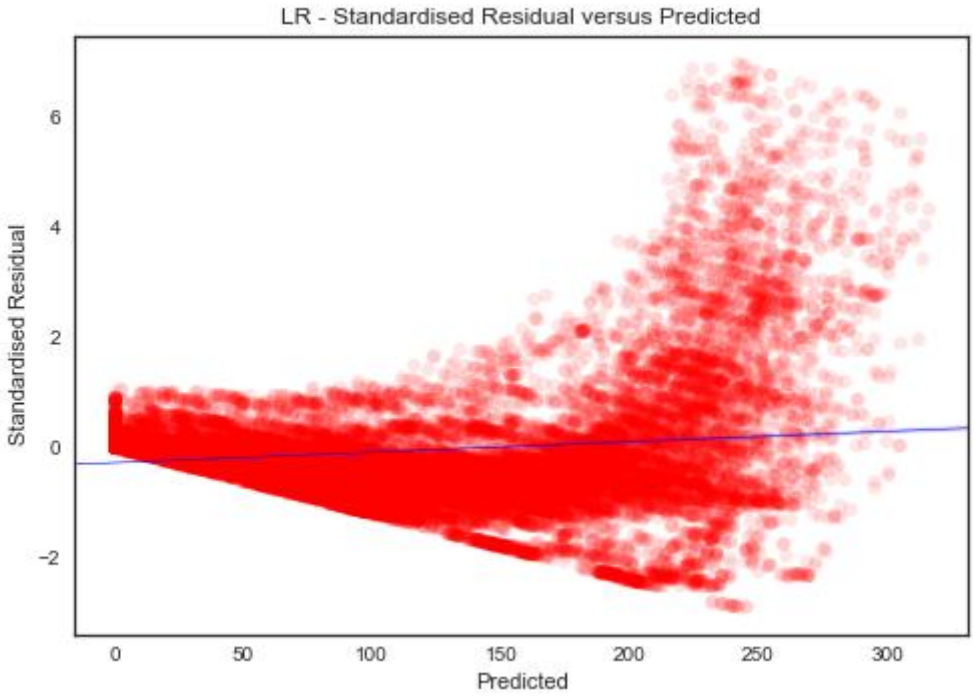

Figure 69: LR - Standardised Residual versus Predicted Time Taken

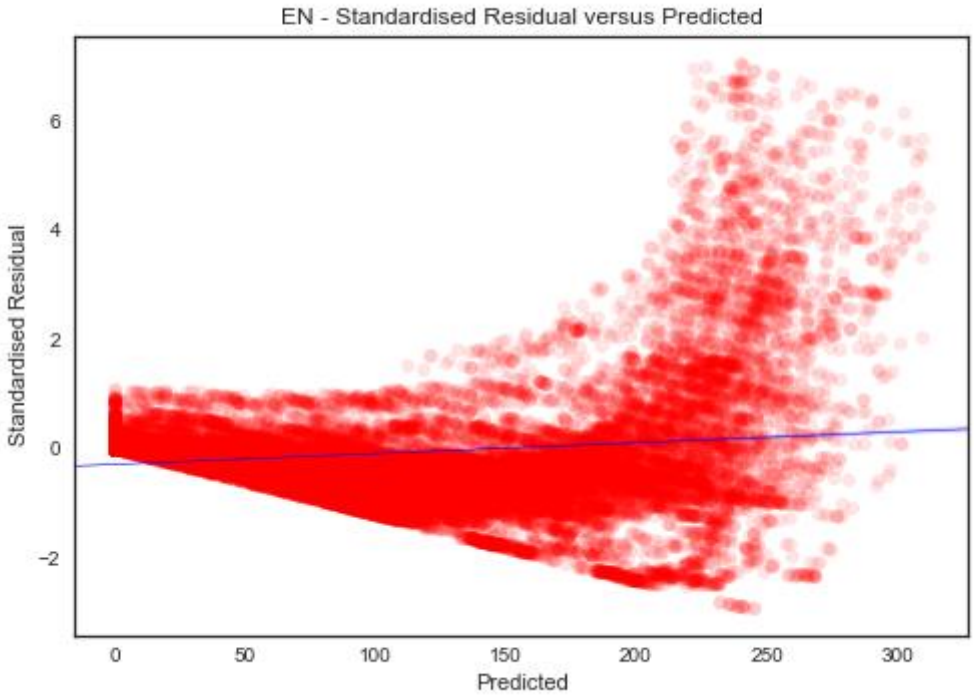

Figure 70: EN - Standardised Residual versus Predicted Time Taken

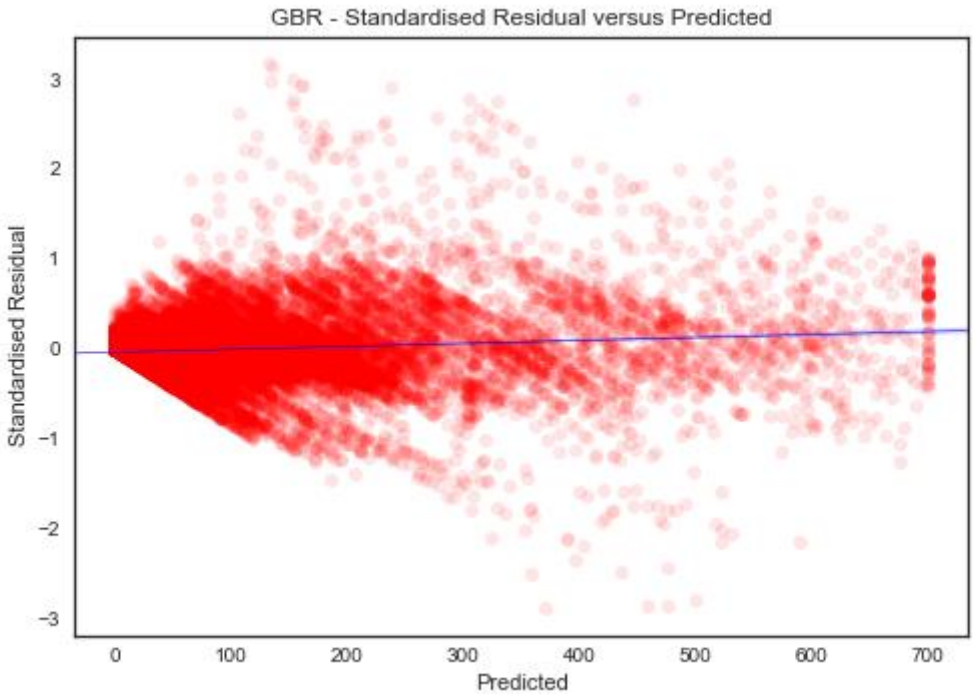

Figure 71: GBR - Standardised Residual versus Predicted Time Taken

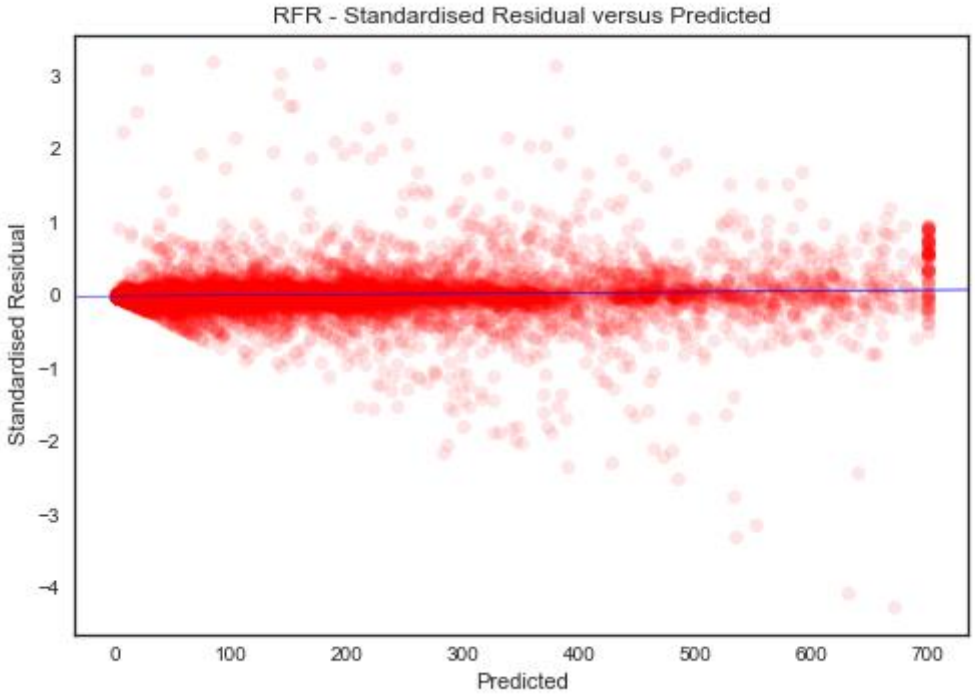

Figure 72: RFR - Standardised Residual versus Predicted Time Taken

Appendix 6: Experiment C1: Testing with unseen June data

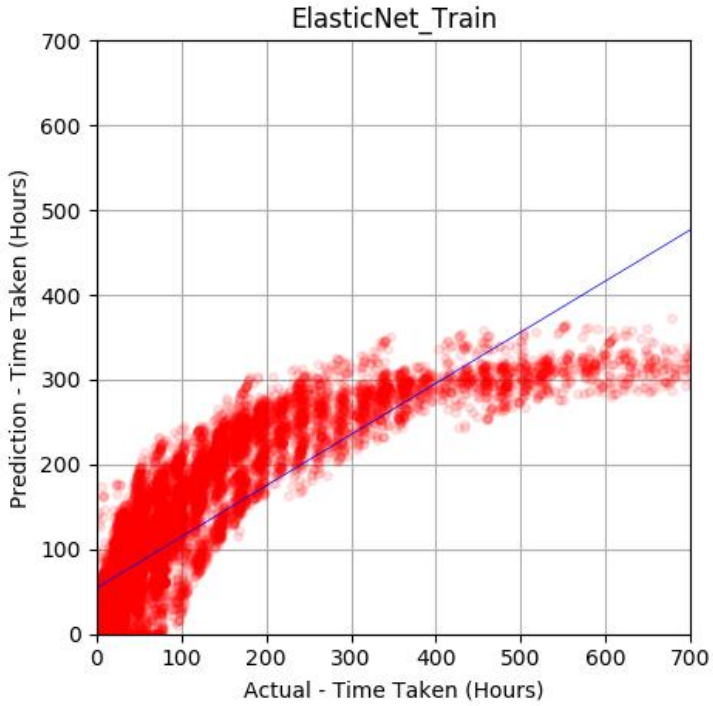

Figure 73: EN Training data - Actual versus Predicted

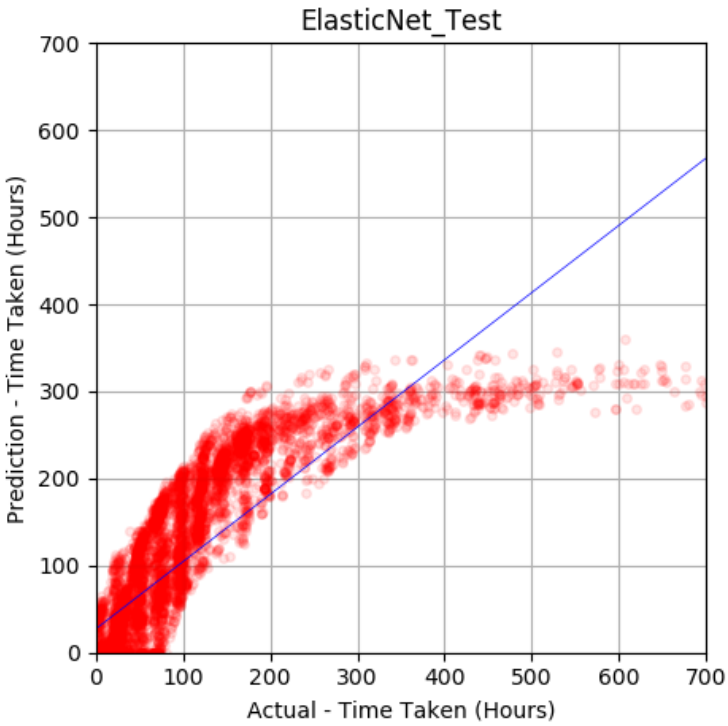

Figure 74: EN Testing data - Actual versus Predicted

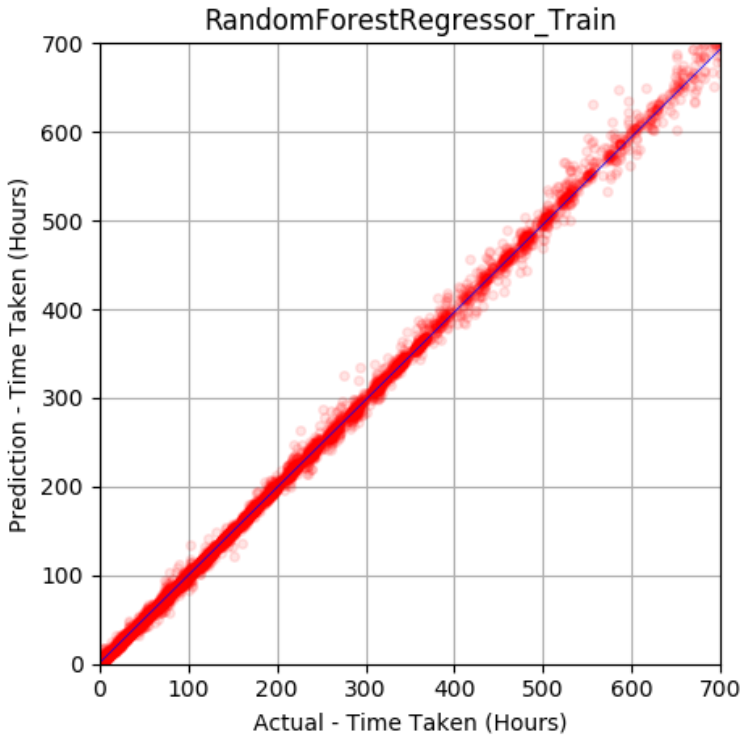

Figure 75: RFR Training data - Actual versus Predicted

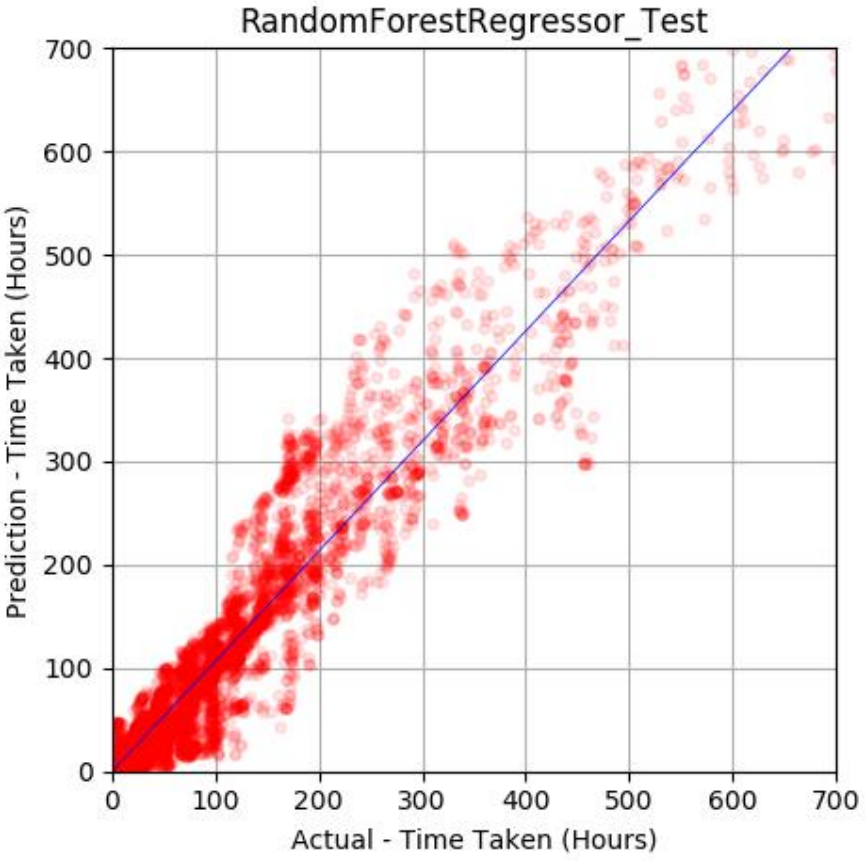

Figure 76: RFR Testing data - Actual versus Predicted

Appendix 7: Experiment C2: Testing with Unseen July Data

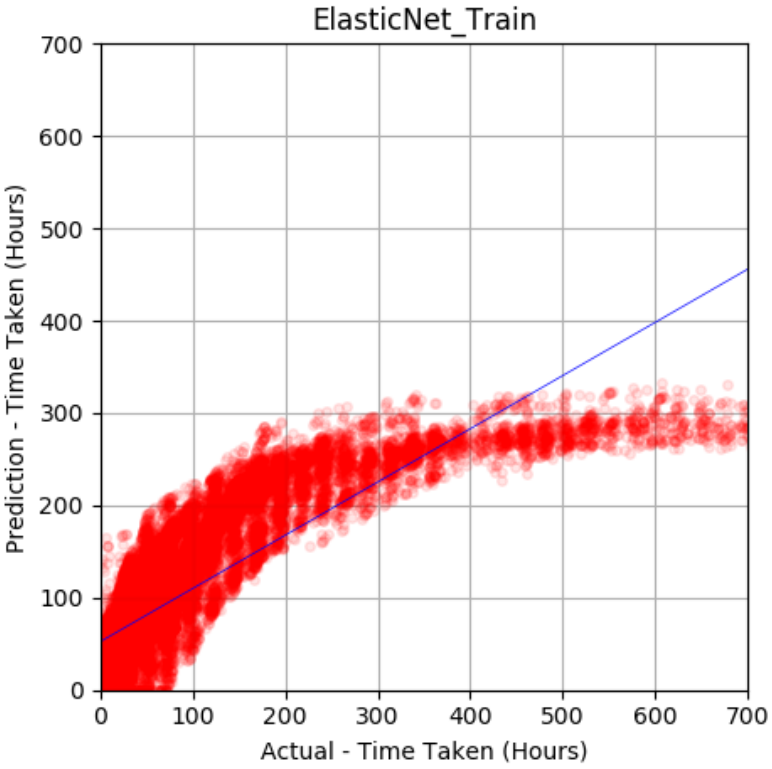

Figure 77: EN Training data - Actual versus Predicted

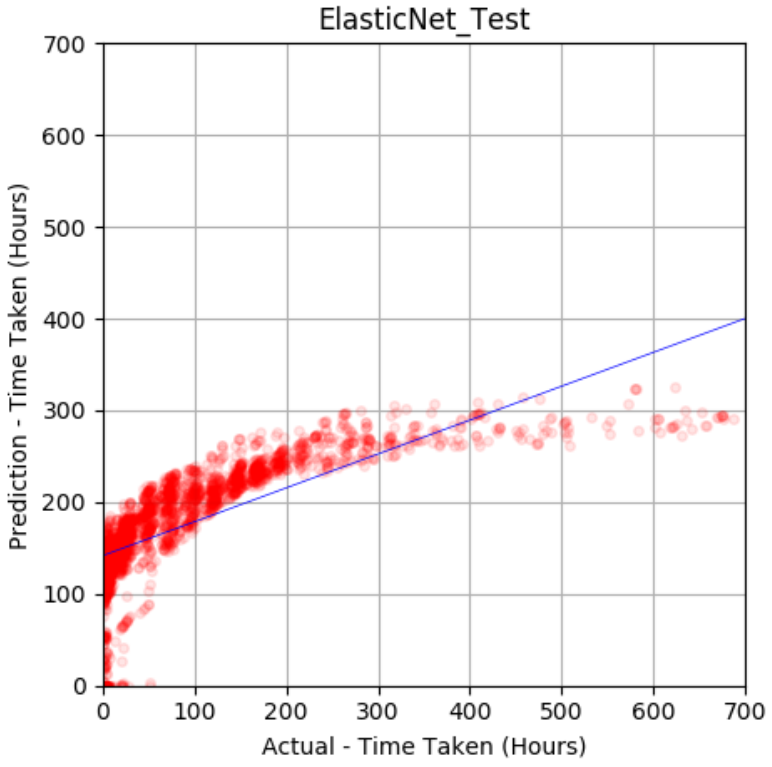

Figure 78: EN Testing data - Actual versus Predicted

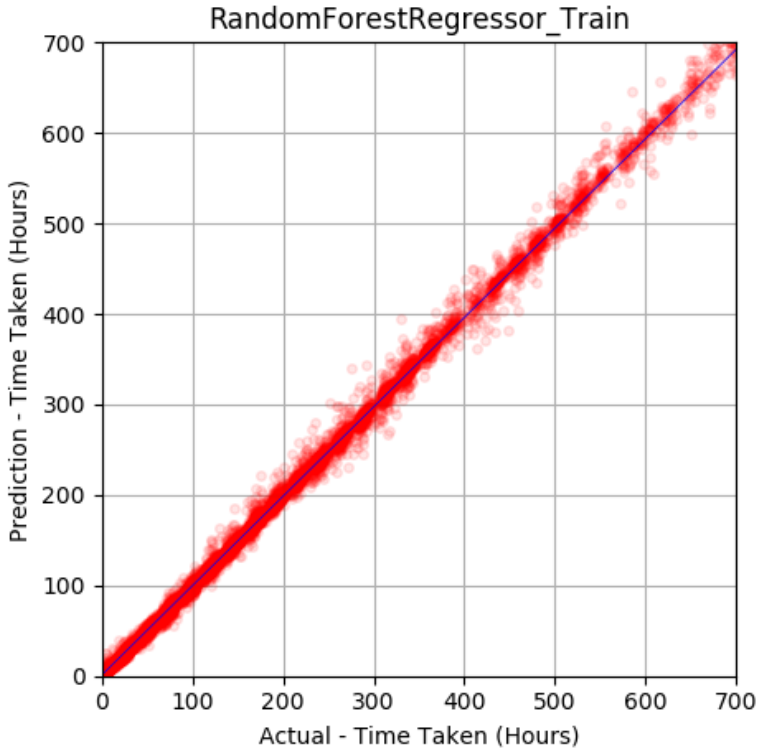

Figure 79: RFR Training data - Actual versus Predicted

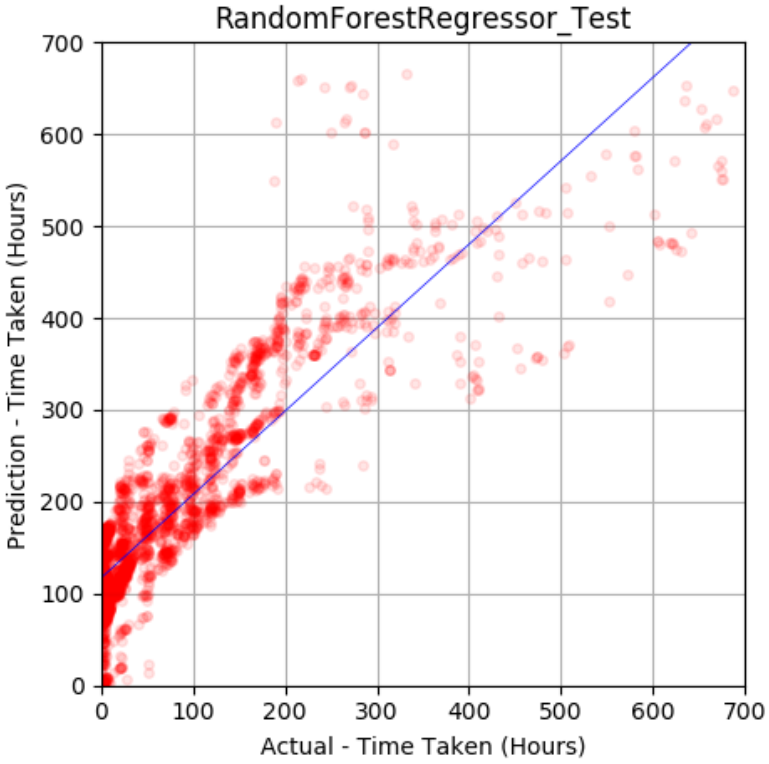

Figure 80: RFR Testing data - Actual versus Predicted

Appendix 8: Test Results as Sample Size Increases (R^2)

Table 11: Test Results for each algorithm as the sample size increases

Test Results	**Dataset (Named by date pulled)**			
R Squared (std)	**20/04/2017**	**23/06/2017**	**25/07/2017**	**15/08/2017**
No. Observations	4583	19924	25986	28204
Baseline	0.0000	0.0000	0.0000	0.0000
LR	0.8129 (+/-0.0377)	0.7550 (+/-0.0092)	0.7194 (+/-0.0069)	0.6708 (+/-0.0142)
EN	0.8190 (+/-0.0123)	0.7429 (+/-0.0122)	0.7091 (+/-0.0074)	0.6590 (+/-0.0114)
RFR	0.9926 (+/-0.0013)	0.9911 (+/-0.0010)	0.9836 (+/-0.0011)	0.9779 (+/-0.0027)
GBR	0.9827 (+/-0.0016)	0.9639 (+/-0.0028)	0.9463 (+/-0.0022)	0.9399 (+/-0.0041)

References

Bergstra, J. and Bengio, Y., 2012. Random search for hyper-parameter optimization. Journal of Machine Learning Research, 13(Feb), pp.281-305.

Breiman, L., 2001. Random forests. Machine learning, 45(1), pp.5-32.

Ceci, M., Lanotte, P.F., Fumarola, F., Cavallo, D.P. and Malerba, D., 2014, October. Completion time and next activity prediction of processes using sequential pattern mining. In International Conference on Discovery Science(pp. 49-61). Springer, Cham.

Chen, T. and Guestrin, C., 2016, August. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794). ACM.

Friedman, J.H., 2001. Greedy function approximation: a gradient boosting machine. Annals of statistics, pp.1189-1232.

Goeman, J., Meijer, R. and Chaturvedi, N., 2016. L1 and L2 penalized regression models. R Foundation for Statistical Computing https://cran. r-project. org/web/packages/penalized/vignettes/penalized. pdf. Accessed, 23.

Günther, C. and van der Aalst, W., 2007. Fuzzy mining–adaptive process simplification based on multi-perspective metrics. Business Process Management, pp.328-343.

Lakshmanan, G.T. and Khalaf, R., 2013. Leveraging Process-Mining Techniques. IT Professional, 15(5), pp.22-30.

Liaw, A. and Wiener, M., 2002. Classification and regression by randomForest. R news, 2(3), pp.18-22.

Microsoft, 2017. Microsoft Logo. [online] Available at: https://www.microsoft.com/en-us/legal/intellectualproperty/trademarks/usage/logo.aspx [Accessed 28 Aug. 2017].

Nakatumba, J. and van der Aalst, W.M., 2009, September. Analyzing resource behavior using process mining. In International Conference on Business Process Management (pp. 69-80). Springer, Berlin, Heidelberg.

Pedregosa et al., 2017, Scikit-learn: Machine Learning in Python, JMLR 12, pp. 2825-2830, 2011.

Pika, A., van der Aalst, W.M., Fidge, C.J., ter Hofstede, A.H. and Wynn, M.T., 2012. Predicting deadline transgressions using event logs. Lecture Notes in Business Information Processing, 132, pp.211-216.

Polato, M., Sperduti, A., Burattin, A. and de Leoni, M., 2014, July. Data-aware remaining time prediction of business process instances. In Neural Networks (IJCNN), 2014 International Joint Conference on (pp. 816-823). IEEE.

Polato, M., Sperduti, A., Burattin, A. and de Leoni, M., 2016. Time and activity sequence prediction of business process instances. arXiv preprint arXiv:1602.07566.

Reijers, H.A., 2007. Case prediction in BPM systems: a research challenge. Journal of Korean Institute of Industrial Engineers, 33(1), pp.1-10.

Riekert, M., Premm, M., Klein, A., Kirilov, L., Kenngott, H., Apitz, M., Wagner, M. and Ternes, L., 2017. PREDICTING THE DURATION OF SURGERIES TO IMPROVE PROCESS EFFICIENCY IN HOSPITALS.

Rogge-Solti, A. and Weske, M., 2013, December. Prediction of remaining service execution time using stochastic petri nets with arbitrary firing delays. In International Conference on Service-Oriented Computing (pp. 389-403). Springer, Berlin, Heidelberg.

Schonenberg, H., Sidorova, N., van der Aalst, W.M. and van Hee, K.M., 2009, June. History-Dependent Stochastic Petri Nets. In Ershov Memorial Conference (Vol. 5947, pp. 366-379).

Shearer, C., 2000. The CRISP-DM model: the new blueprint for data mining. Journal of data warehousing, 5(4), pp.13-22.

Tibshirani, R. 1996, "Regression Shrinkage and Selection via the Lasso", Journal of the Royal Statistical Society. Series B (Methodological), vol. 58, no. 1, pp. 267-288.

Van Der Aalst, W., 2012. Process mining: Overview and opportunities. ACM Transactions on Management Information Systems (TMIS), 3(2), p.7.

Van Der Aalst, W., 2012. Process mining: Overview and opportunities. ACM Transactions on Management Information Systems (TMIS), 3(2), p.7.

Van Der Aalst, W., Adriansyah, A., De Medeiros, A.K.A., Arcieri, F., Baier, T., Blickle, T., Bose, J.C., van den Brand, P., Brandtjen, R., Buijs, J. and Burattin, A., 2011, August. Process mining manifesto. In International Conference on Business Process Management (pp. 169-194). Springer, Berlin, Heidelberg.

Van der Aalst, W., Pesic, M. and Song, M., 2010. Beyond process mining: from the past to present and future. In Advanced information systems engineering (pp. 38-52). Springer Berlin/Heidelberg.

Van Der Aalst, W.M., 2013. Business process management: a comprehensive survey. ISRN Software Engineering, 2013.

Van der Aalst, W.M., Reijers, H.A., Weijters, A.J., van Dongen, B.F., De Medeiros, A.A., Song, M. and Verbeek, H.M.W., 2007. Business process mining: An industrial application. Information Systems, 32(5), pp.713-732.

Van der Aalst, W.M., Schonenberg, M.H. and Song, M., 2011. Time prediction based on process mining. Information systems, 36(2), pp.450-475.

Van der Werf, J.M.E., van Dongen, B.F., Hurkens, C.A. and Serebrenik, A., 2008, June. Process discovery using integer linear programming. In International Conference on Applications and Theory of Petri Nets (pp. 368-387). Springer, Berlin, Heidelberg.

van Dongen, B.F., Crooy, R.A. and van der Aalst, W.M., 2008, November. Cycle time prediction: When will this case finally be finished?. In OTM Confederated International Conferences" On the Move to Meaningful Internet Systems" (pp. 319-336). Springer Berlin Heidelberg.

Wikipedia, 2017. CRISP-DM Process Diagram. [online] Available at: https://en.wikipedia.org/wiki/File:CRISP-DM_Process_Diagram.png [Accessed 28 Aug. 2017].

Yerkes, R.M. and Dodson, J.D., 1908. The relation of strength of stimulus to rapidity of habit-formation. Journal of comparative neurology, 18(5), pp.459-482.

Zou, H. and Hastie, T., 2005. Regularization and variable selection via the Elastic Net. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 67(2), pp.301-320.